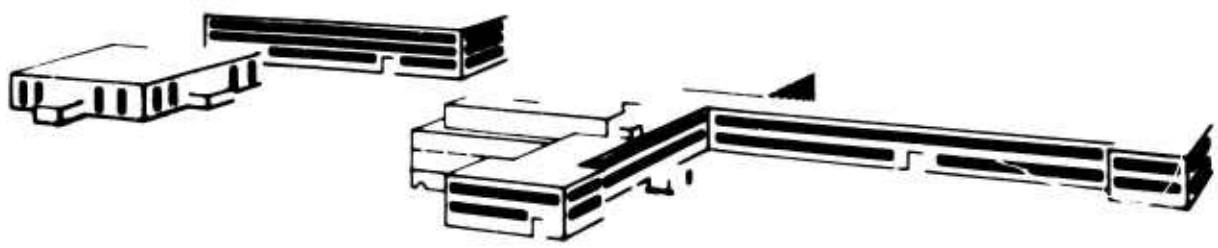
AD623590

352.14-R-1

AN INFORMATION RETRIEVAL SYSTEM MODEL

CLEARINGHOUSE
FOR FEDERAL SCIENTIFIC AND TECHNICAL INFORMATION
Hardcopy | Microfiche
$4.00 | $1.00 | 144 pg.
ARCHIVE COPY



HRB-SINGER, INC.
SCIENCE PARK · STATE COLLEGE, PENNSYLVANIA

HRB-SINGER, INC.
A SUBSIDIARY OF THE SINGER COMPANY
Science Park, State College, Pa.

352.14-R-1

AN INFORMATION RETRIEVAL SYSTEM MODEL

Contract Nonr 3818(00)

October 1965

Copy No. 12 of 95 Copies

Prepared by:

Charles R. Blunt

Reviewed by:

Kenneth F. Barber
Methodology Branch

Contributers:

Richard M. Malinoski
H. Dean Wilson

Approved by:

Ralph C. Stevenson
Behavioristics Laboratory

Submitted by:

C. Richard Conger
Project Director

Reverse (Page ii) Blank

ABSTRACT

This report presents some work pertinent to the quantitative evaluation of information retrieval (IR) systems and extends the development of an Information Retrieval System Simulation Model. The work is sponsored by The Information Systems Branch of the Office of Naval Research and is part of a program whose major objective is to formulate general purpose simulation models of the various functional components found within intelligence systems.

The present IR system simulation examines system response time, equipment/personnel utilization and idle time, and delay time in queue. The general IR model is, essentially, an ordered grouping of basic retrieval functions. The nature of the functions and the configuration of the system can be specified to the simulation program by the investigating engineer. The simulation program can ultimately be used by Naval planners of information retrieval systems to evaluate alternative IR configurations, to identify and illustrate the need for an IR system, and to assess the effectiveness of such a configuration.

Response time of an IR system is cited as one necessary criterion of system performance and is closely related to the operating costs, another quantitative measurement of an IR system.

Reverse (Page iv) Blank

FOREWORD

An ability to measure and evaluate intelligence system performance in various situations is very useful and desirable. A capability to predict intelligence system performance under different situations is, however, much more useful and desirable. Consider, for example, the difficult problem in answering a most basic question "Can an information retrieval system support and improve intelligence operation X?" If the answer to this question is in the affirmative, the system planners and design engineers must continue and ask questions such as.

(1) What type of system can best support X?

(2) What will be the operating costs?

(3) Under what conditions will the input/utilization load saturate the system?

(4) How flexible is the system to meet possible changes in operation X?

(5) What personnel/equipment is necessary to maintain 100% operation?; 95% operation?

(6) What are the effects in the system if component A is changed.

(7) What system configuration will provide the fastest response? What configuration will provide the most economical effective support? What configuration will provide the most reliable support?

These, and an almost endless list of similar questions, must be answered to some extent in examining possible intelligence retrieval configurations.

The work presented in this report is one of several efforts geared to providing Navy planners and engineers with an evaluation tool that will provide insight into the predictable behavior of information retrieval systems in intelligence operations.

Reverse (Page vi) Blank

HRB

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont'd)

LIST OF ILLUSTRATIONS

Reverse (Page x) Blank

I. INTRODUCTION

> "An information system is...a complex of people, equipment and procedures working together to provide needed information to a group of users."
>
> Charles P. Bourne

One facet of the project work under contract Nonr 3818(00) is the use of computer simulation techniques to evaluate information retrieval (IR) configurations found in various intelligence systems. The evaluation of an information retrieval capability is another step in the modular approach towards the evaluation of intelligence systems. Figure 1 illustrates the relationship of this work with some of the present, past and future efforts, e.g.,

(1) previous development of a collection model provides a method of estimating the likelihood of collecting various target data,

(2) present and past efforts in the development of an intelligence system output model provide a method of assessing the utility of the collected data with respect to the intelligence product requirements,

(3) the information retrieval model is to eventually provide a measure of the time and cost expended to manipulate the data base in answering the intelligence questions.

Thus, if the collection effort is evaluated to satisfy the intelligence needs, then the retrieval effort within the intelligence system can be evaluated with respect to response time and operating costs. Eventually, other evaluation criteria such as capacity and enhancement of processed information with respect to accuracy and validity will be considered.

Simulation of the information retrieval capability links the formal expression of an information need (query) with the probable collection of data (data base). The energy required to answer a question is a function of both the capability of the retrieval system and the relationship between the data base and the question. A sketchy or noisy data base, for example, could

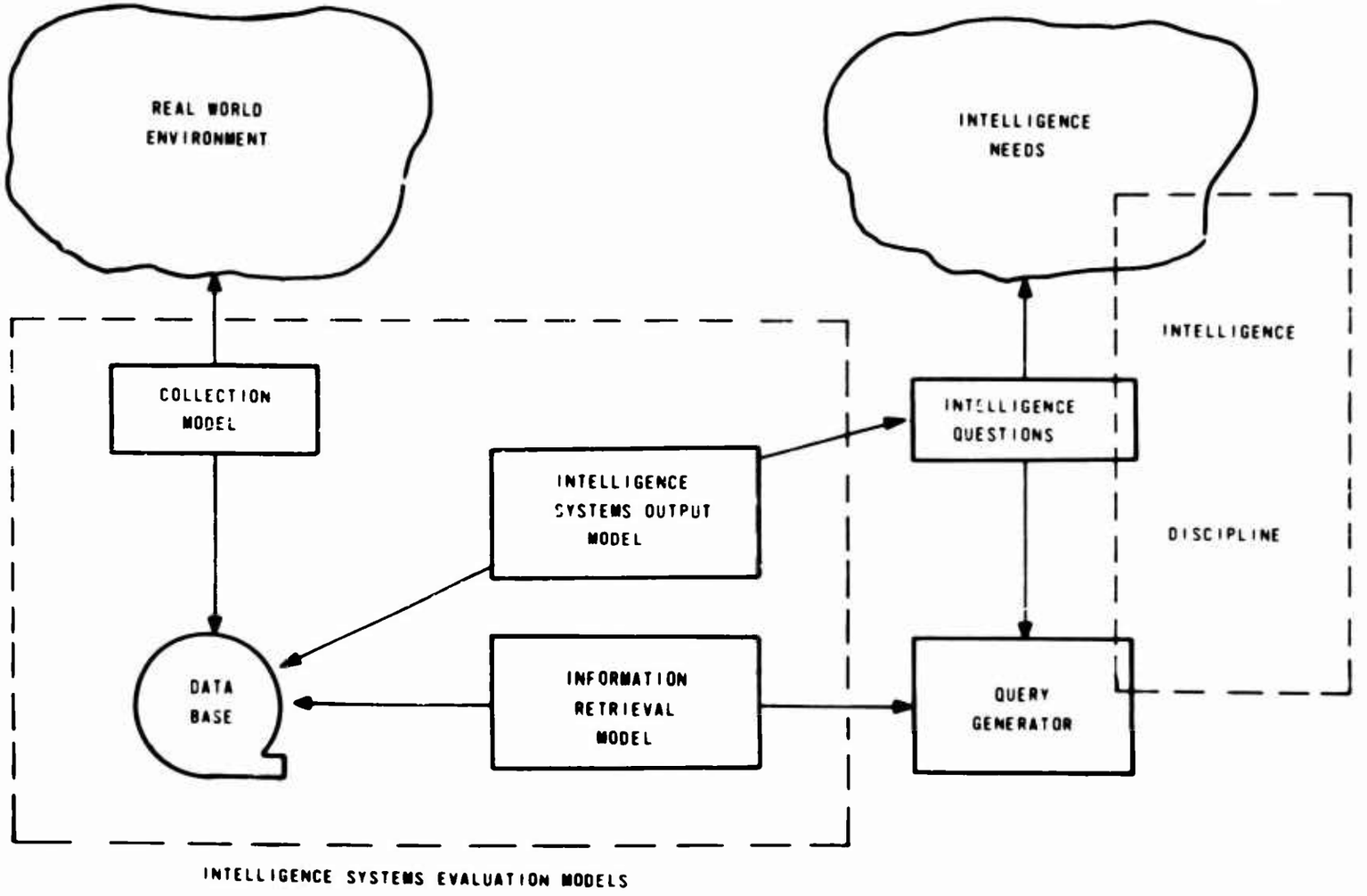


FIG. 1 RELATIONSHIP OF MODEL DEVELOPMENT IN INTELLIGENCE SYSTEMS EVALUATION

impose complex requirements upon the formal translation of a question into a query set. The basic question "Where is the enemy?" could generate a complex set of queries seeking both well identified units and events that may possibly infer the whereabouts of the remaining elements. The contents and validity of the data base, therefore, tend to influence how an intelligence question is posed to a storage and retrieval system. It is anticipated that query formulation can be performed by a Query Generator, operating upon the classes of intelligence questions and tempered by the nature of the sensory output.

The Information Retrieval Model is not restricted to the evaluation of Intelligence Systems. There are many information systems employing IR methods and techniques that can be examined through this tool. Moreover, the information retrieval simulation program can aid the system designer by providing a mechanism for testing alternate system configurations.

Initial considerations of the Information Retrieval research effort have been presented in a previous report.[1] This report continues from that discussion and gives

(1) an expansion of the initial considerations,

(2) the present general model of computer based IR systems,

(3) an outline and examples of the simulation program, and

(4) a discussion of present plans for continuing work effort.

The appendices of this report give (1) a study of the effect of errors in the specification of parameters, (2) the instruction listing of a newly developed program to integrate the effect of parallel operations in the IR model and (3) an annotated bibliography of research reports pertinent to simulation and evaluation studies of IR systems.

---

[1] HRB-Singer report 352-R-17 "The Simulation and Evaluation of Information Retrieval Systems," April 1965, (AD 464619).

A. BACKGROUND AND RATIONALE

With few exceptions, every field of endeavor is threatened with being inundated with increasing amounts of data. Current collection methods and equipment provide intelligence analysts with an almost steady stream of data; support groups can provide operational commanders with large volumes of facts and figures concerning enemy status, projected EOB's and trend analyses, analysts provide (hence, seek) an expanding wealth of technical information. The crisis today is not necessarily the lack of pertinent data, but instead, the selection of these data from the available pool. Within the intelligence community this crisis produces a strange dilemma; i.e., modern warfare has significantly reduced the necessary lead-time margin, hence, the intelligence analyst now has less time to examine more data than ever before.

Mechanized information storage and retrieval concepts are continually being proposed as solutions to the information crisis. It is contended, for example, that the high-speed large-memory capability of computers can aid data selection, organization and analysis; that ADP techniques can aid in reducing the time necessary in producing reliable intelligence estimates. Experience has shown, however, that there may be a long costly road between a design concept and an operating system. In the opening remarks to an assembly gathered to discuss the problems of testing and evaluating information retrieval systems, it was stated that "...history has shown that only one in five of the R&D efforts produces a successful operating system."[1] More recently, Business Week pointed out that planners and engineers are having skeptical second thoughts concerning the value of computers for the storage and dissemination of scientific and technical information.[2] There are two probable reasons why reasonable concepts have been frustrated by unsuccessful development. First -- there is a wide range of complex significant problems confronting system development. Second -- there is a serious lack of testing techniques that can provide constructive early feedback to the design engineers.

---

[1] Opening address by Dr. [illegible]kos G. Photias at the Seventh Institute of Information Storage and Retriev[illegible], sponsored by the American University, 1-4 February 1965.

[2] "Research Briefs," Business Week 24 July 1965; p. 106.

At present, one of the more successful (although expensive) methods of testing the feasibility of a concept is to build a pilot configuration for operational experimentation. In this manner, representative problem areas are probed and the findings serve as feedback to the continuing testing and development effort. Computer simulation could provide design engineers with more timely information (at less cost) than is now available from operational experimentation if the simulation can effectively hit the problem areas. The following discussion briefly explores some considerations for the simulation of information storage and retrieval systems.

1. Problems That Need Solutions

In the development of an information storage and retrieval system, there are some basic problems that must be solved if the system is to be successfully implemented. High ranking among the critical problems are those that directly relate to satisfying the system user's requirements. Dr. Taube offers as a necessary condition for the test and evaluation of information systems the examination of "energy" necessary to produce desired output from the total file.[1] This energy can be measured by

(a) system response time,

(b) quality of presentation, and

(c) cost of operation.

These examination criteria do not exclude the necessity of other studies (e.g., input validation, utility of products from system, etc.), but do point to aspects of the system that are important to performance evaluation. It should be noted that the first and third criteria are interrelated and are quantitative measures of performance.

[1] "Necessary and Sufficient Conditions for Evaluating Information Systems," paper delivered by Dr. Mortimer Taube at the Seventh Institute of Information Storage and Retrieval.

a. System Response Time

From a systems user's perspective, system response time is the period that lapses between the statement of information need and the reception of output satisfying this need. Response time requirements may be expressed in fraction of seconds, minutes, hours or days depending upon how the data are to be used. For example, almost instantaneous response is required of tactical data systems providing amphibious assault information; the latest estimate of enemy deployment is frequently required in minutes, and complete order of battle data may be needed in support of an analysis operation within hours of a mission notice. There also exist response time limits on systems operating outside of the intelligence environment, e.g., command-control systems must provide air sector activity within minutes; logistic systems should respond with inventory level data within hours and referral centers should provide response within several days. Response time is a function of equipment speed; efficiency of the man-machine interface, operating programs and procedures and the communications capability within the system.

b. Quality of Presentation

There is a wide spectrum of output capability corresponding to the wide range in mechanized information systems. System output can be textual or graphic; it may present references to documents or provide abstracts, documents, extracts or pictures; or it may respond with answers to a limited and predetermined set of questions. Assuming that an output capability has been selected that meets with the user's requirements, there is still a severe problem affecting that quality of the presentation.

An effective system should be sensitive to a user's information needs. A request should be answered with a complete output of relevant information. Since the ASLIB-Cranfield Project, attempts have been made to measure the sensitivity of retrieval systems by determining Relevance and Recall ratios.[1] A problem in these studies, however, is the fact that "relevance" has not been

[1] See Evaluation of Indexing Procedures and Retrieval Effectiveness; Project SHARP report June 1964, NAVSHIPS 250-210-3. Also, A Case in the Application of Cranfield Evaluation Techniques, Herner, Lancaster and Johanningsmeir, Herner and Company report delivered 30 August 1964.

reasonably and well defined. Consider, for example, two users with the same request. An output from the system may be relevant to one but not to the other because he has already obtained those data from other sources. There is a need to distinguish between relevance to a request and relevance to the user. A user's need may be dynamic, what is relevant now may not be on a later request; what is not now considered relevant may become critical in a later search. Goffman and Newill make such a distinction by defining a relevant set to be ". . . that subset of the file the user would have selected as a response to his query had he searched the file himself. . . . A subset of the file which is appropriate to the need is called a pertinent set."[1] These definitions, however, imply that (1) when the user searches the file he may not recognize data appropriate to his need (otherwise, he would always select pertinent data) or (2) the user will be bound by the logic of his query and will not select data appropriate to his needs outside the scope of the query. In this latter case, any logician could replace the user without altering the effect of the search. In either case, there is no clear consistent manner by which one may classify file data with respect to each request.

There seems to be a general agreement that sufficient pertinent output with minimal noise is good. If this quality is defined as system effectiveness, then effectiveness is a judgment and not a quantifiable representation of system performance. This does not deny the utility of Cleverdon's ratios, but does point to a limitation in their application, i.e.,

(1) a user may not know the complete nature of his problem; hence, will not know his complete information needs as he initially approaches the system,

(2) a user may not be able to adequately express those needs he does recognize,

(3) data within the system may be relevant to the problem but not to the user's needs as he may already have exhausted those sources.

---

[1] Methodology for Test and Evaluation of Information Retrieval Systems, W. Goffman and V. A. Newill, Comparative Systems Laboratory Technical Report No. 2, July 1964, p. 7.

A "ratio" evaluation of system effectiveness may not reflect these considerations.

Within an intelligence system, information retrieval is frequently in support of the problem solving activities of the intelligence analyst. If the intelligence analyst can "solve" (to any extent) a problem with the data found within the data base without using an IR system, then the energy expended in utilizing the IR system to reach the same solution is one measure of the effectiveness of the IR system. Ideally, efficient use of the IR system should improve the analyst's performance at least to the point where

(1) he can derive the "same" solution in less time, or

(2) he can derive a "better" solution in the same time.

Unfortunately, it is extremely difficult to validate an intelligence estimate, hence, a "better" solution may often only look different -- thus, the effectiveness of an intelligence IR system may also be a matter of judgment.

The Intelligence Systems Output Model will provide an estimate of the likelihood that the collection effort will collect data pertinent to some given problem. If we assume a nondegrading input transformation, then this estimate will be the likelihood that these data are in the data base of the IR system. If we also assume a nondegrading transformation of the intelligence question into a set of queries, then the retrieval effort will produce these pertinent data with the same likelihood.[1] The energy required to produce these data is a function of

(1) the number and nature of the queries,

(2) the structure and size of the data base,

(3) the equipment and personnel characteristics and

(4) the IR system configuration.

[1] These assumptions are normally not valid. Degredations in the transformations will usually increase the energy used in the retrieval operation. These transformation functions, however, lie outside the scope of the present work effort. They will be considered in future efforts within the systems evaluation tasks.

This energy can be measured in terms of response time and operating costs and can be determined with simulation techniques.

c. Cost of Operation

The operating cost of information systems is the sum of the operating cost of each function (e.g., data collection, input preparation, storage, retrieval and presentation) plus the maintenance and support costs incurred to maintain the operations. These costs can be associated with equipment, personnel, facilities and material. Sometimes the operating costs may also include initial costs prorated over several years. Initial costs may include expenditures for research, development, equipment purchase and personnel training.

Cost determination is a reasonably straightforward accounting of expenditures. Value determination, however, is a more complex problem. The value of a system and its costs are not necessary in proportion nor are they measurable in the same manner. In an information system, costs can be denoted at every stage of processing from collection to output. Benefits, however, result only from actions utilizing these outputs. The value of an information system is, therefore, connected with user performance and capability which may only be assessed in a qualitative manner.

2. First Research Objective

The first objective in this research effort is to simulate the response time of mechanized information storage and retrieval systems. This initial goal has been selected for the following reasons.

a. Information retrieval is a vital aspect of intelligence information systems.

b. Response time is a quantitative measure of system performance.

c. Effective response time simulation can be easily modified to provide operating costs of retrieval.

d. Effective general information retrieval simulation can provide engineers with data concerning the time-cost effects obtained with different mixes of equipment, personnel and procedures.

While it is recognized that response time and costs are not sufficient conditions for the evaluation of an information system, they are necessary conditions. This first research objective is expected to provide basic data necessary for the development of an engineering tool that can aid design, test and development efforts.

B. OVERVIEW OF THE IR MODEL

Development of the information retrieval response time model has proceeded by first examining a selected domain of systems, constructing an outline of the functions that are performed in these systems and considering how the operating time of the functions can be depicted. Three basic aspects that illustrate this model development are,

(1) the initial assumptions about the kinds of systems to be simulated,

(2) the functions and interrelationships that have been included within the basic model, and

(3) the techniques to be provided to determine processing time for each operation within the retrieval simulation.

These aspects are briefly touched upon in the following discussion and are given to provide both an overview of the IR model and a base for subsequent material presented within this report.

1. Initial Assumptions

The present information retrieval model is intended to reflect fundamental aspects of a mechanized system's response to a user's request for data. In the present stage of development, several assumptions have been made that restrict the scope of the model; hence, may limit the real world domain of systems reflected. This limitation is intentional and has been made in order to expedite the development and validation of the basic model. Once the basic model has been successfully tested, model expansion can be considered with respect to the utility and limitation in applying simulation results.

The following assumptions are explicit in the present model development effort.

a. Information retrieval is accomplished with computer processing.

b. The user interacts with the system at only two points; i.e., he poses a question and receives an answer -- he does not monitor intermediate processing.

c. The system is responsive to one user at a time; i.e., it is not a time sharing system.

These assumptions essentially focus attention upon the simulation of computer based information service systems and do not consider on-line time sharing systems.

2. Functions and Interrelationships

Figure 2 gives a simple flow diagram of the basic model logic. In this diagram, each rectangular block represents a time consuming function. The block labeled PROCESS QUERIES represents the most complex function within the basic model and will be discussed in the later chapters within this report. It should be noted that a distinction is made between "question" and "query." A question is the user's expression of information need; a query is the formal expression of the search statement. A question may generate several queries which may be posed against different files. The "answer" from the system is the sum of the output from these input queries.

There are three decision points within the basic model that enable an engineer to establish a sensitivity level for the system (or system concept) to be simulated. The setting of a sensitivity level is accomplished by determining (or estimating) the probability that a question will be rejected, that it will be asked again and that the answer will be acceptable to the user. Therefore, if it is possible to "measure" relevance and pertinence and express these attributes in a quantitative manner, then these decision points can be used to imbed them within the simulation. If these attributes are not expressible in a quantitative manner, a judgment or an experimental value can be used to set some sensitivity level for the system. While such a level may not be exact or truly representative

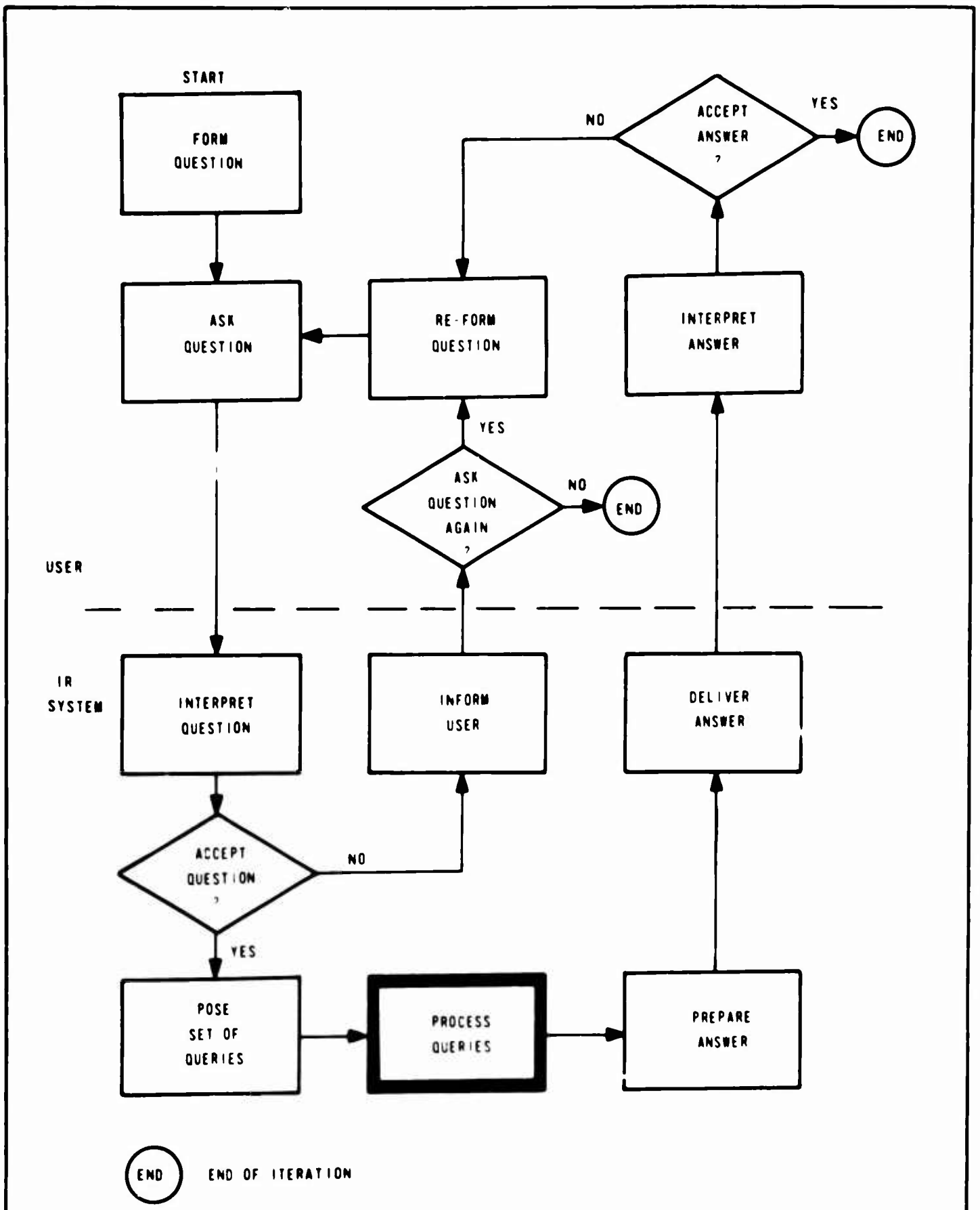


FIG. 2 BASIC MODEL LOGIC

of an operating system, the ability to alter the acceptability of system response can provide useful insight into the effects of user dissatisfaction with system output. Within the present basic model logic, failure to accept the answer penalizes the system by increasing the operation time (hence cost) for the processing of the question. Given a system configuration, different processing response times will be obtained by varying the sensitivity level for the system. Experimentation with the simulation may thus provide threshold values for measuring user acceptance, e.g., if the user rejects more than X% of the output, the processing effort per question becomes too costly.

3. Processing Time

System response time for a given question is calculated by (1) determining the time required in each step of the operations necessary to answer the question and (2) determining what amount of time is lost by delays encountered in processing.

a. Required Operating Time

Two methods are provided to determine operating times of the different steps required in the processing of a question. These are:

(1) time distribution tables

(2) processing characteristic time formulae.

The simulation program will allow a system engineer to use either method in determining the operating times of a given step. Moreover, the program will accept combinations of both methods in expressing the information retrieval system.

Time distribution tables would normally be provided through experimental observation.[1] A step in the processing operation (e.g., search file) would be sampled and an approximating curve (see Figure 3) constructed denoting the relationship between time expenditure and the number of test runs in

[1] See HRB-Singer report 352-R-17; pages 30-36.

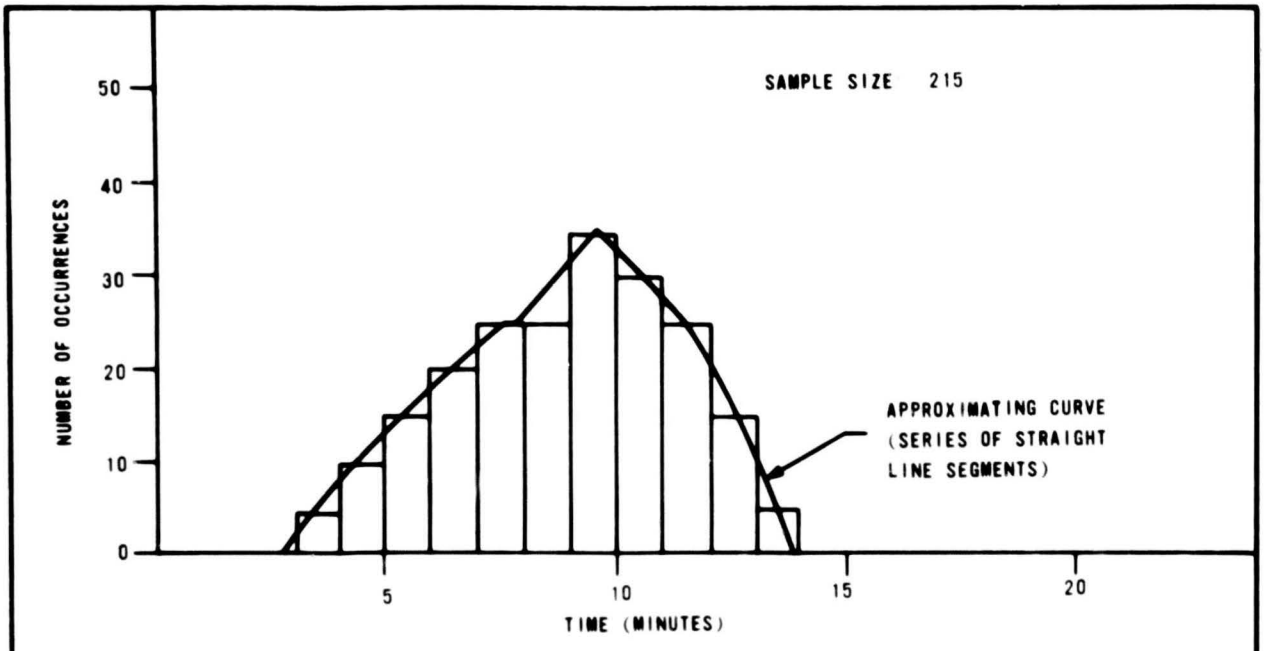


FIG. 3 APPROXIMATING (PDF)

the sample. This curve would then be used as an approximating probability density function (pdf) for the operating time of the processing step. There may exist different time distribution tables for a given processing step, these corresponding to different classes of questions or queries being processed.

Processing time formulae are now being developed to calculate operating time by examining the characteristics of the operation. For example, the time required to read a file is a function of four variables, i.e.,

(1) file size,

(2) file organization,

(3) search strategy, and

(4) equipment speed.

Formulae, relating the general variables of an operation to the time expenditures of processing, will provide design engineers with a capability to depict or represent specific equipment configurations within the operating

system. Moreover, this capability would enable an engineer to project the specifications of new equipment into a design and measure the change in the operation.

b. Delay Time

Delay time in processing occurs when a needed component is not available. This situation may happen when the needed component is busy with some other task or is down for repair. Presently, the IR model considers the effect of several queries flowing through a processing configuration of personnel and equipment. Within this flow, two or more data units may be at the same point for servicing at the same time. In this event, a queue will form and one or more data units would encounter a delay in processing. The development of the simulation program now provides a first-come-first serve queue unloading strategy. Continuing effort, however, will provide for optional strategies, e.g., selecting the data unit with the shortest servicing time, consideration of what flow will minimize idle time at the central processor, etc.

Calculation of delay time caused by equipment down-time is not presently included in the simulation program. This facet of processing delay may be added in the near future when the topics of operational reliability and maintainability are examined.

Reverse (Page 16) Blank

II. PROGRAM DEVELOPMENT

> "A model has many useful roles. In one of these, it serves to expose the fundamental nature of any given retrieval problem and helps to clarify the basic assumptions of the system designed to meet that problem. It is thus a device for mirroring a problem and an instrument for shaping its solution."[1]

The simulation program will have three fundamental parts; these are conveniently identified with three basic subroutines:

(1) the Event Sequence Generator,

(2) the Sequence Integrator, and

(3) the Data Analysis Programs.

Within the framework of the Event Sequence Generator, an engineer will be able to define the processing events and the behavior of the data flow within the system to be simulated. The subroutine will, for every iteration, (1) identify the events required in the processing of each query, (2) denote the sequence that these events are performed and (3) determine the time used in each processing event. The output from this subroutine provides the input for the Sequence Integrator.

The Sequence Integrator considers the effects of parallel and simultaneous operations in the retrieval process, e.g., integrating the effects of query one being processed by the central processor while query two is being prepared. This subroutine (1) allocates equipment and personnel to the different processing events, (2) calculates the delay time of each query at each processing step and (3) calculates the amount of idle time for each component (equipment and personnel) during information retrieval.

---

[1] Study of Theories and Models of Information Storage and Retrieval; Report No. 1: Problems, Systems and Methods, Donald J. Hillman, August 3, 1962, (AD 282084), p. 22.

The Data Analysis Programs operate upon the simulation output data which is produced during each iteration of the model. These programs reduce this output data and summarize the results. Included in the analysis summary are

(1) number of iterations considered,

(2) average response time per iteration,

(3) time variance and standard deviation,

(4) time histogram data (response time versus number of occurrences, frequency of occurrence and cumulative frequency of occurrence),

(5) operating time of each system component,

(6) delay time at each step,

(7) component idle time during processing.

Additionally, simulation parameters (e.g., number of queries generated, answers rejected, etc.) will be summarized.

A. EVENT SEQUENCE GENERATOR

This subroutine generates the operating time and sequence of processing events for each query treated in the response time simulation. The processing events considered in the present model are

(1) Pose query - includes conceptual process of forming the query and the physical process of preparing a query form.

(2) Send query for processing -- dispatch of query to processing facility.

(3) Convert query for entry -- conversion into machine medium.

(4) Check conversion -- verification or check of conversion.

(5) Send converted query for entry -- dispatch of converted query to the computer room.

(6) Computer processing -- all operations involving the central processor, e.g., query entry, file search, output editing, on-line printing.

(7) Off-line printing -- output display of selected material. Does not involve use of central processor.

(8) Return query for correction -- errors in preparation of query may cause query to be rejected and returned for correction.

These eight events represent time consuming operations in the processing of a query. All eight events, however, are not necessarily applicable in every IR system. Moreover, in some situations, some events may be required more than once in the processing of a query.

The following subsections describe (1) a method of classifying the queries to be selected in the simulation processing, (2) techniques that an engineer can use to regulate the model to reflect an IR system, and (3) basic considerations in event time expression.

1. Selection of Queries for Processing

Prior to entry into the Event Sequence Generator subroutine, the simulation program will have selected some question type, determined the time required to formulate and ask the question and will have generated a set of query types to be used in the retrieval effort (see Figure 4). Mechanically, these simulation aspects will be accomplished by using a random number generator to select some element (e.g., question type) from a

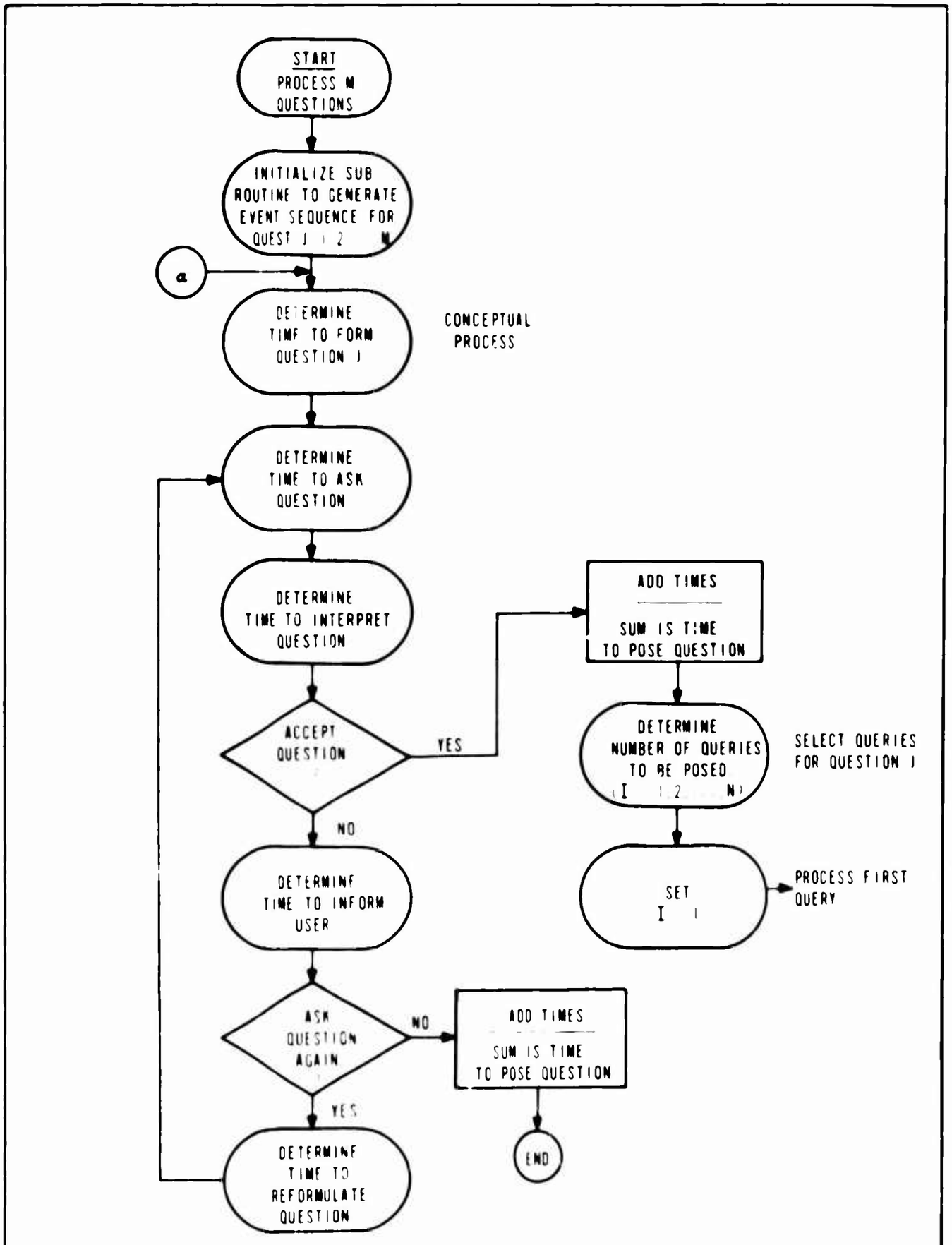


FIG 4 START OF SIMULATION

distribution table.[1] The more important aspect of this technique centers upon how well the IR system can be depicted by such distributions.

A basic contention of this research effort is that the questions that are to be posed against a given IR system can be reasonably categorized. Moreover, these question categories can be used to depict the nature of the retrieval effort. For example, an intelligence information retrieval system may be required to be responsive to problems concerning

(1) activity within geographical areas,

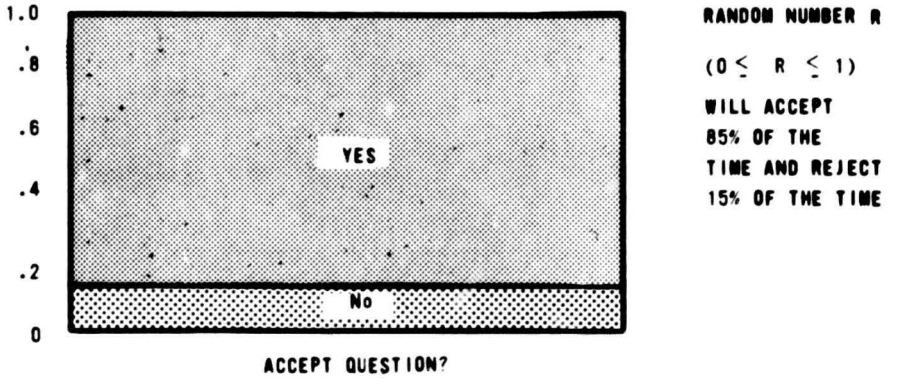
(2) activity within time frames,

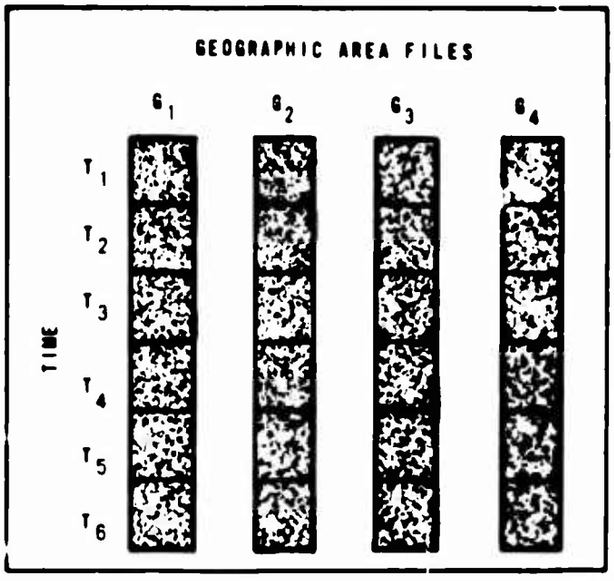
(3) unit history and disposition.

The data base, therefore, may be structural into area files, these being maintained in chronological sequence to expedite retrieval, e.g.,

---

[1] The decision points to "accept the question" or "ask the question again" are also passed with the aid of a random number generator, e.g.,





The kinds of questions that can be posed against these files can be generated by considering the possible combinations of the variables defined in the file format (see Figure 5).

FILE VARIABLES:
- ACTIVITY (A)
- GEOGRAPHIC AREA (G)
- TIME FRAME (T)
- UNIT IDENTIFICATION (U)

CATEGORY				EXAMPLE
U	A	T	G	WHAT UNITS ENGAGED IN TRIALS WERE OBSERVED DURING JANUARY, 1965, IN AREA ONE?
U	A	T		WHAT UNITS WERE AT SEA IN JUNE, 1964?
U	A		G	WHAT UNITS WERE BUILT IN AREA TWO?
	A	T		HOW MANY SPACE SHOTS WERE ATTEMPTED BETWEEN 1 OCTOBER 1964 AND 31 JANUARY 1965?
	A			WHAT ATOMIC TESTING HAS BEEN NOTED?
U		T		WHERE WAS UNIT X BETWEEN 4 APRIL 1964 - 10 JUNE 1964?

ETC.

FIG. 5 GENERATION OF QUESTION CATEGORIES

It may be judged that not all generated categories are applicable. These would be given a zero expected frequency of selection and eliminated from consideration. The remaining categories would be given some positive expected frequency notation (e.g., UAT - 40%; UT - 15%, AT - 8%, etc.).

It is possible to further subdivide these categories into specific query types by considering that each variable may take on different values. For example, the variable UNIT may represent identifiable objects such as vessels, organizations, facilities, installations, personalities and aircraft. Each object, in turn, may be identified by some combinations of distinctive items, e.g.,

VESSEL

Type	Class	Pendant	Name	Country
SSN	SKATE	584	SEADRAGON	U.S.
MSF	AUK	377	QUAIL	U.S.
DE	DEALEY	1014	CROMWELL	U.S.
CVAN		65	ENTERPRIZE	U.S.

Retrieval may be made against a specific unit (e.g., SSN SKATE) or against a set of units (e.g., SSN U.S.). In general, each question category may represent a complex of variable values that depict the range of queries that could be generated by each question. Figure 6 illustrates how a question category could be subdivided into specific query types. Again, not all generated query types are necessarily significant or applicable.

One criterion of significance is the pragmatic consideration of what differences are created in processing by different queries. In the example of four chronological geographic area files, the query represented by variable values

(TYPE, CLASS) & (YEAR)

would have to be placed against all four files; whereas the query

(TYPE, CLASS) & (YEAR) & (G_1)

would be placed against only one file. Moreover, since the files are arranged in time order, the query

QUESTION CATEGORY UNIT TIME AREA

UNIT

VESSELS	TYPE	CLASS	PENDANT	NAME	COUNTRY	...
ORGANIZATIONS						
FACILITIES						
INSTALLATIONS						
PERSONALITIES						
AIRCRAFT						

AND

TIME

YEAR	MONTH	DAY	HOUR

AND

AREA

G_1	PLACE NAME	COORDINATE	GRID	BODY OF WATER
G_2				
G_3				
G_4				

EXAMPLES

1. TYPE PENDANT AND YEAR MONTH AND BODY OF WATER
2. CLASS AND YEAR AND G_1
3. TYPE CLASS AND YEAR MONTH AND G_1 OR G_2
4. TYPE COUNTRY AND YEAR MONTH DAY AND G_3

FIG 6 GENERATION OF QUERY TYPES

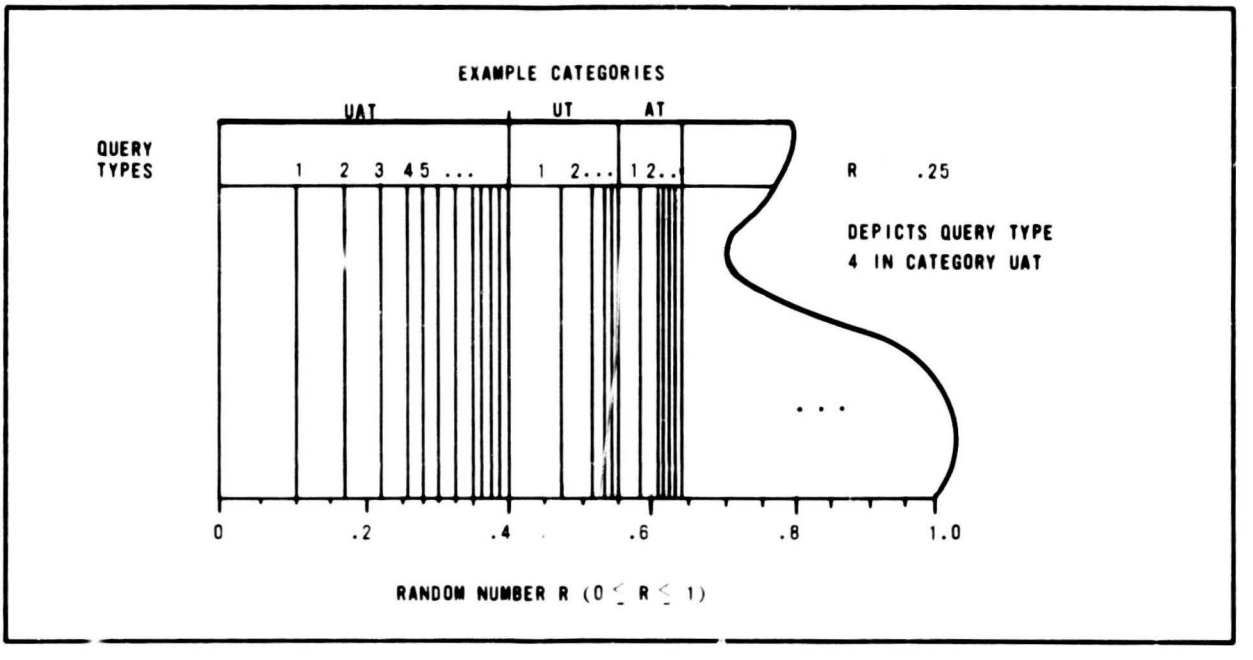
(TYPE, CLASS) & (YEAR)

may have to examine more file records than the query

(TYPE, CLASS) & (YEAR, MONTH, DAY)

since the more specific query could terminate file search once the stated month has been exhausted.

Once distinctive query types have been identified within each category, a conditional expected frequency of selection can be assigned to each type. Thus, if category X has been chosen, there is some probability that query type x_i, within this category, will be selected. These expected frequency of selection values can be estimated from an analysis of the user's information requirements. The values can then be used in establishing a distribution table depicting the question and queries to be levied in the simulation effort, e.g.,



This distribution, in turn, determines the nature of the load placed against the simulated IR system.

2. Determination of Processing Steps

Once the nature of the queries to be processed in an iteration have been determined, the Event Sequence Generator subroutine determines the processing steps (and the time expended in each step) of each query. This determination of processing flow is governed by three facets of the simulation program, i.e.,

(a) the processing events and configuration of the IR model,

(b) a provision to bypass processing events within this model, and

(c) a provision to reject and return queries for corrective action.

The second and third facets provide the engineer with a capability to tailor the model to fit the IR system being simulated.

a. Configuration of Processing events

Figure 7 illustrates the eight processing events now contained in the IR model. These events do not necessarily reflect any given system; but, instead, are intended to reflect the general functions that may be performed in the retrieval process. Following is a brief discussion illustrating how these functions can be viewed in terms of actual system components.

(1) POSE QUERY -- personnel analyze question and formulate query or queries necessary to the selection of pertinent data from the system. This activity may include the preparation of query forms or formal request statements.

(2) SEND FOR PROCESSING -- if the processing facility is located some distance from the request point, the query may be dispatched, transmitted or phoned for processing.

(3) CONVERT FOR ENTRY - the query is converted into a machine recognizable language and placed into a machine medium. This may be accomplished with card punch or tape punch equipment, optical or mark sense

scanners or special devices such as direct query consoles and data recorder equipment.

(4) CHECK CONVERSION -- the conversion of the query is checked for accuracy. This may be accomplished with card or tape verification equipment, manual scanning or display-editing devices. If the conversion is checked, there exists some probability that query will not be accepted and will be returned for correction or reconversion.

(5) SEND FOR ENTRY -- frequently the computer installation is separated from EAM and other processing equipment; hence, the converted query may have to be dispatched for entry into the central processor.

(6) PROCESS QUERY -- this includes all operations involving the centeral processor, e.g., query entry, file search, record comparison, edit, data summary, etc.

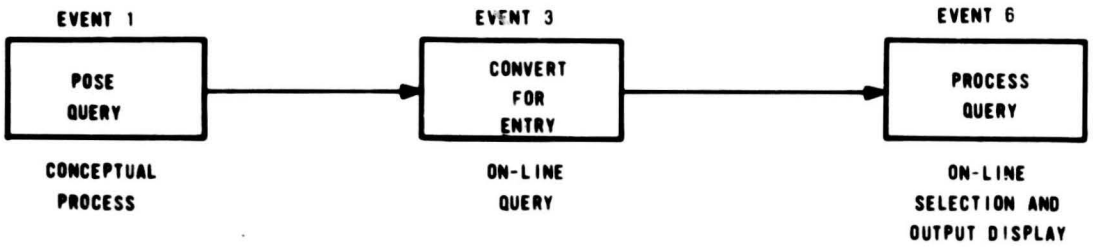
(7) DISPLAY OUTPUT OFF-LINE -- if on-line data display has not been selected, then the central processor may be released after it has transferred the output data onto some output medium (such as magnetic tape). These data may now be displayed by off-line operations such as printing or console viewing.

(8) RETURN FOR CORRECTIVE ACTION -- a query may be rejected at several points in the computer processing. For example, upon entry, the computer check program may detect some error in the query statement or format; upon completion of the file search, a failure to select any data may indicate an error in the search statement; during processing, a read-write redundancy check may indicate a bad tape or malfunctioning processing unit. Problems such as these may abort the present processing attempt and cause the query to be returned (or stopped) for corrective action.

It is not expected that all eight functions are required in the processing of every query of every system. They should, however, provide a general framework within which an engineer can identify time consuming events in the retrieval process.

b. Capability to Tailor Model

As an engineer identifies and defines the functions performed in the retrieval process, he may decide that he does not require all of the events provided in the IR model. For example, the system may use direct entry remote query consoles with on-line remote printers. The engineer may desire to simulate this system under the following configuration:



On the other hand, another system may have different processing paths depending upon the priority or nature of the request. Figure 7 illustrates a hypothetical system having two basic paths depending upon the priority of the request. In this system, a low priority request could pass through events 1, 2, 3, 5, 6, and 7; or if an error was encountered in processing, the steps could be 1, 2, 3, 5, 6, 8, 3, 5, 6 and 7. A high priority request would pass through events 1, 3' and 6 or perhaps 1, 3', 6, 8, 3' and 6.

Within the simulation program, it is possible to always,in effect, bypass some given event by defining the time expenditure for that function to be zero. Also, it is possible to sometimes bypass an event by defining a time distribution (or time expression) having both zero and nonzero values. Moreover, it is possible to key the path of a query to some selected parameter (e.g., high or low priority). For example, in selecting some query type, another distribution table can be examined to determine priority. A high priority query would, in the system depicted in Figure 8, set zero processing times for events 2, 4, 5 and 7. Additionally, a query parameter can also key different time distributions or expressions within the simulation; hence, some query types may be set to process faster than ot query types.

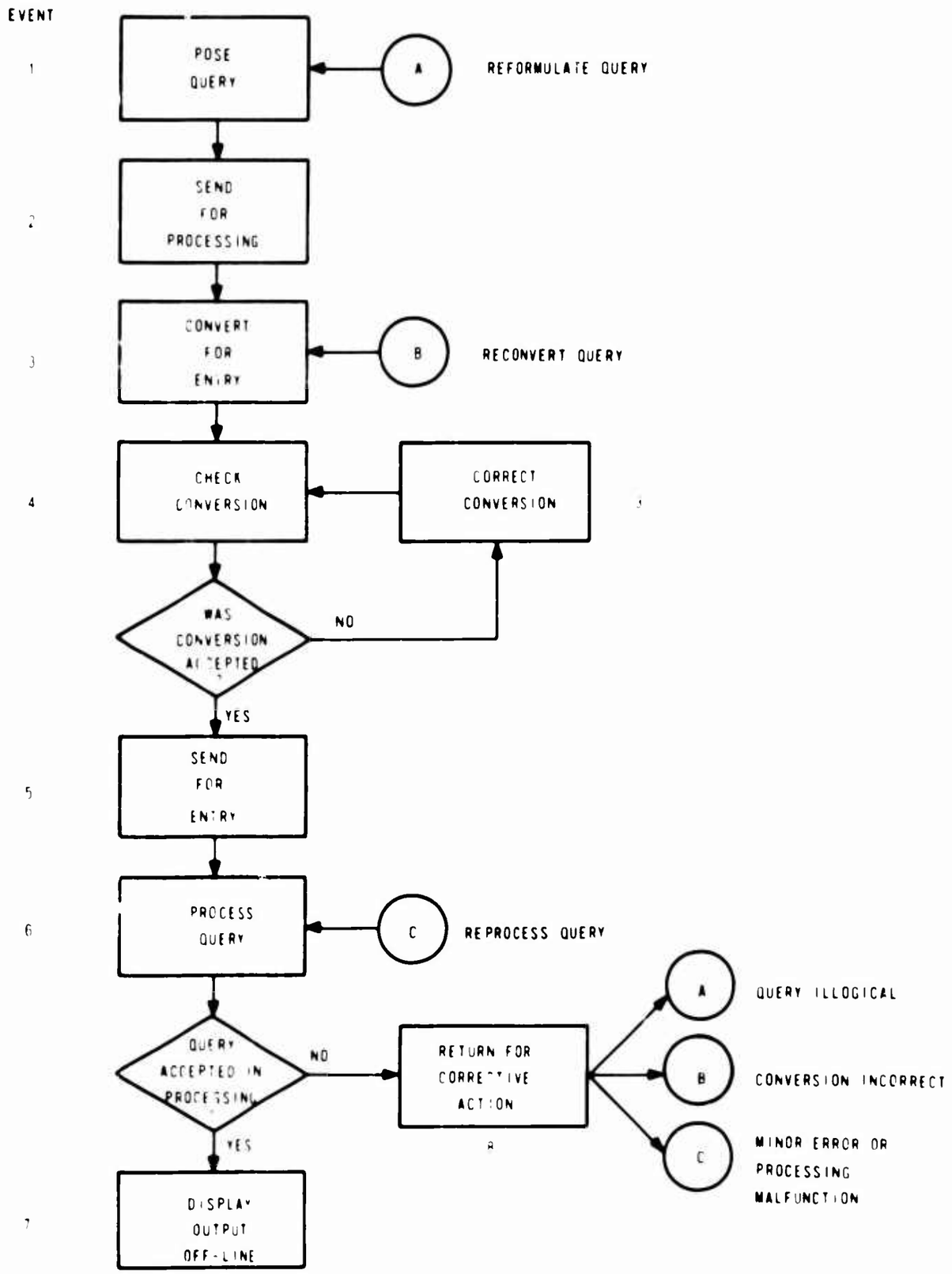


FIG 7 PROCESSING EVENTS

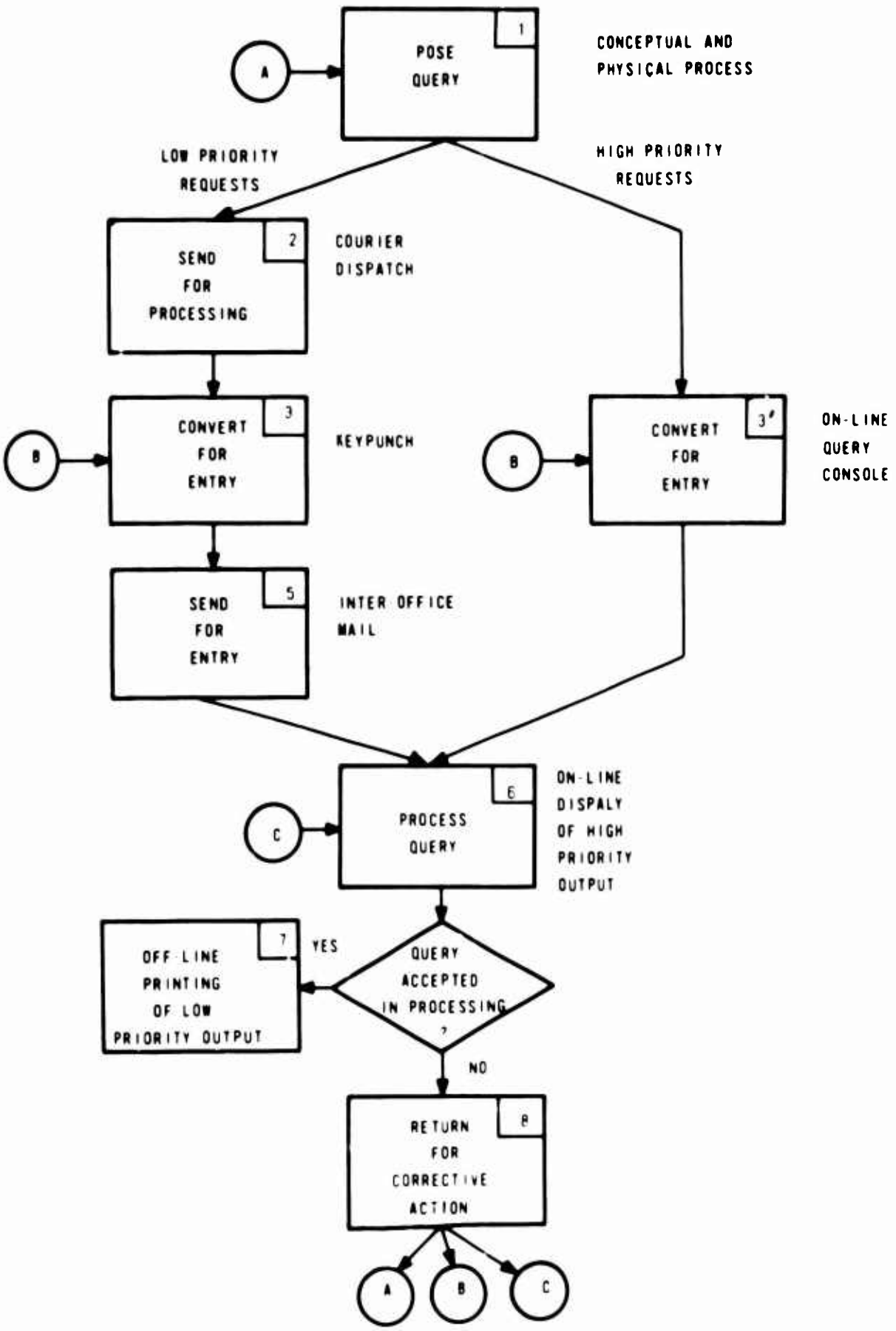


FIG. 8 CONFIGURATION OF HYPOTHETICAL IR SYSTEM

Another capability to tailor the general model outline is provided in an ability to relable some of the events. If, for example, a single communications channel was used to transmit a query for entry (event 5) and return a rejected query for correction (event 8); then by calling both functions EVENT 5, the total channel use time and delays caused by busy channel can be easily calculated.

The decision points to accept or reject the query at different steps and the routing of the rejected query for corrective action are regulated by decision tables. These tables are constructed by the engineer and can be made by determining (or estimating) the probabilities of the different alternatives. Again, a random number generator is used to select a path for each query.

Since the path of a query is determined by a stochastic process, the events and the number of steps required in retrieval will unfold during the operation of the Event Sequence Generator. This "history" is preserved and passed on to the Sequence Integrator subroutine for further processing. In general, each question may precipitate several queries and each query may require a different sequence of processing events. Preservation of processing history, therefore, requires a variable storage area. Figure 9 illustrates such

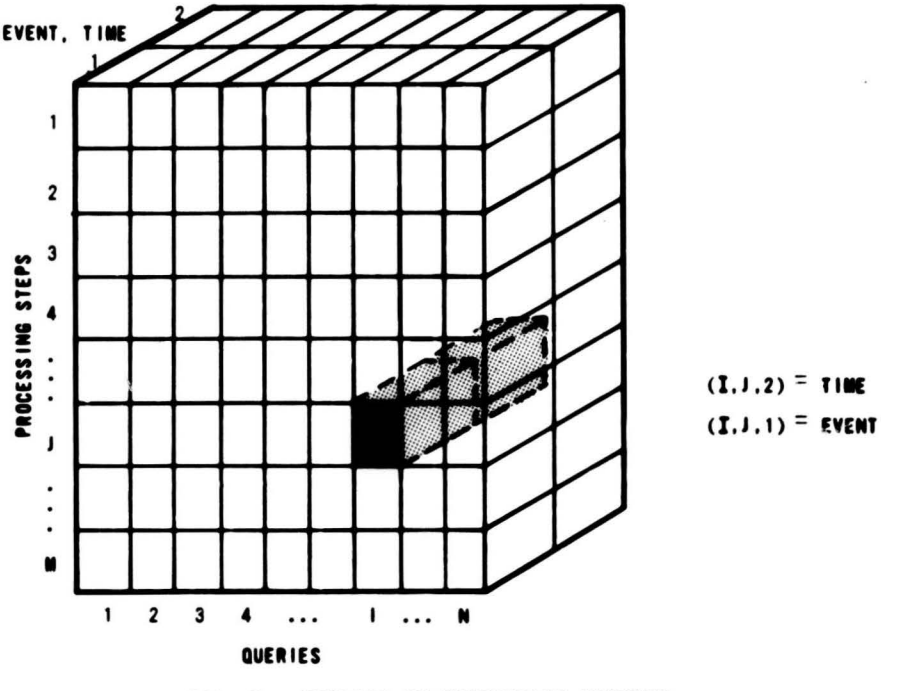
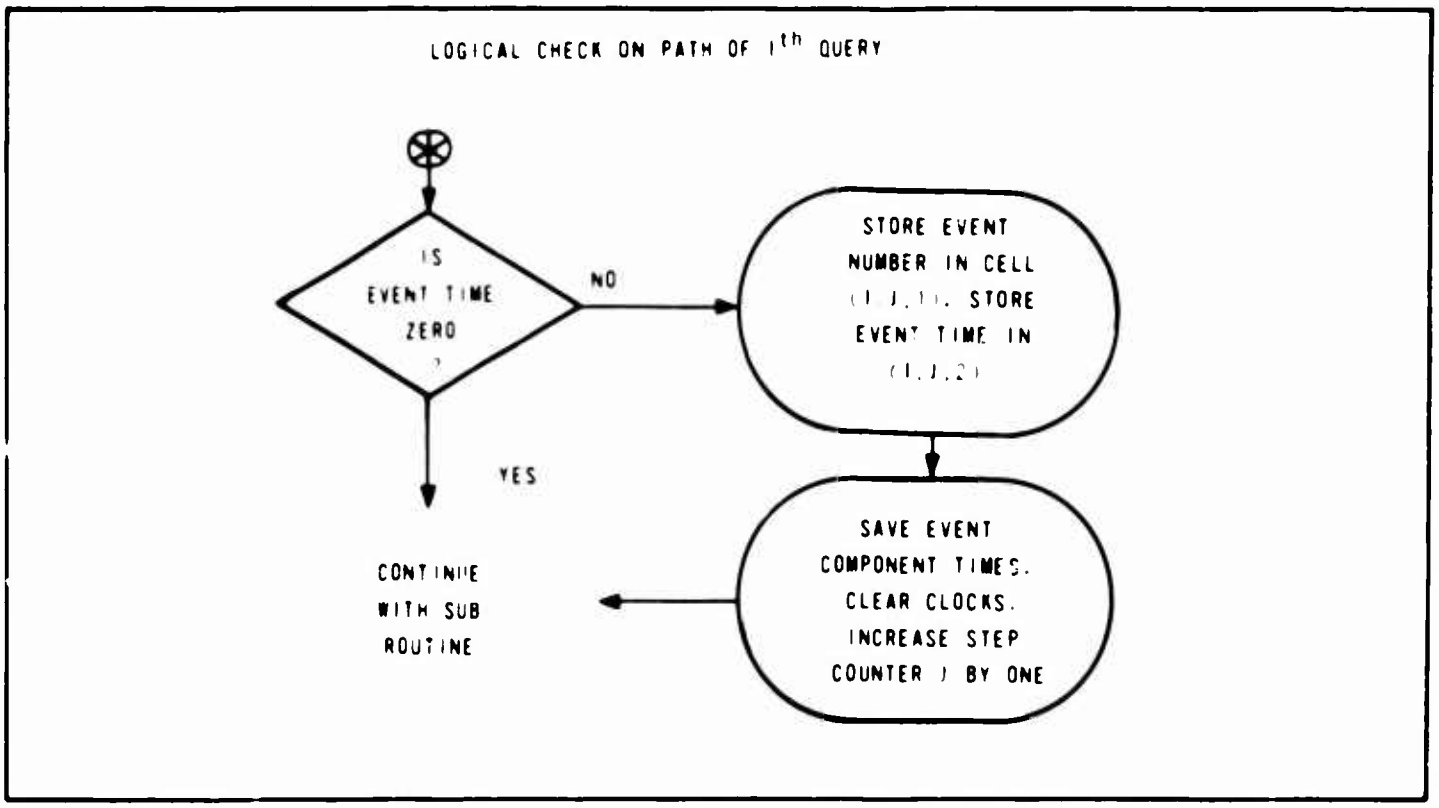


FIG. 9 STORAGE OF PROCESSING HISTORY

a variable storage area as a three-dimensional matrix. Each $(I, J)^{th}$ position contains two cells, i.e., the time consummed by the J^{th} step of the I^{th} query and the number of the processing event.

Each query lamina is filled during an interation of the Event Sequence Generator by a logical check routine following each event, i.e.,



The sum of the event times in each query lamina represents the total processing time required for the query. Delay time will be determined in the Sequence Integrator subroutine.

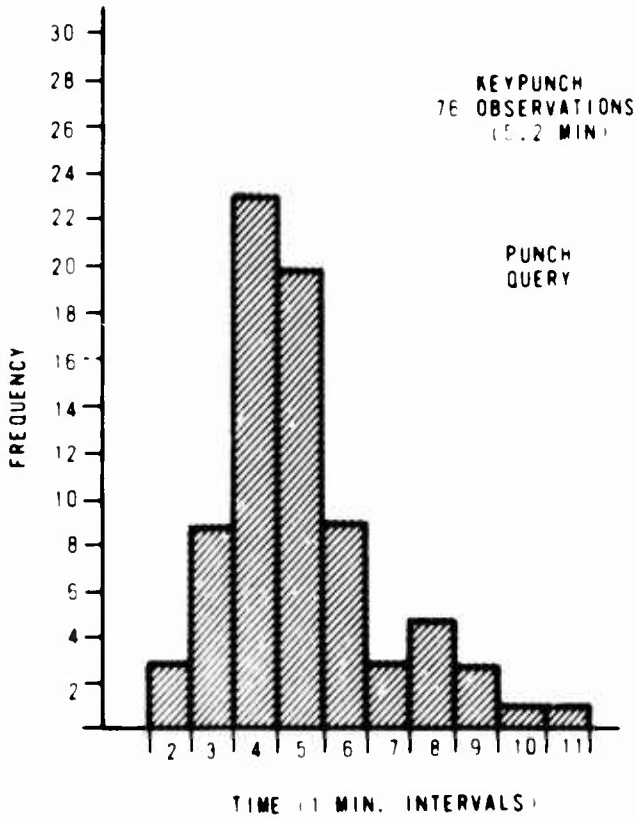
3. Event Time Expressions

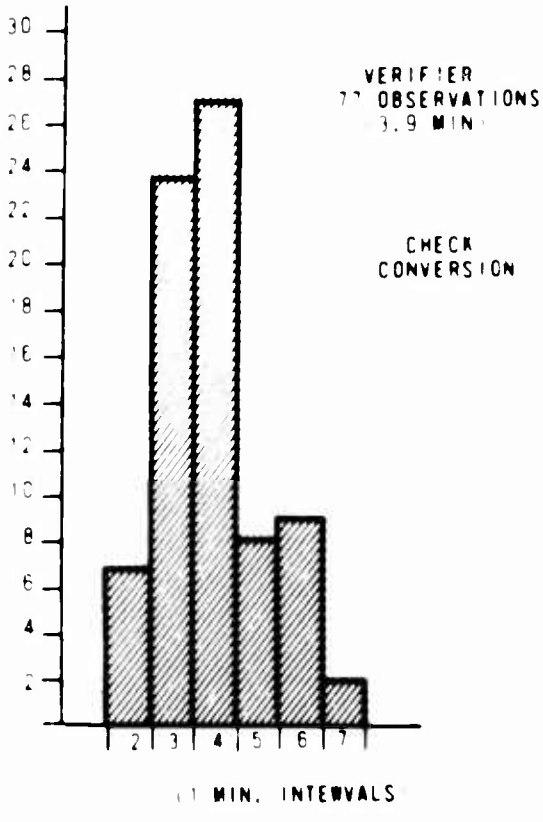
In general, each processing event within a real operating system uses some operating time. Within the simulation program, two methods are provided to represent these time expenditures, i.e.,

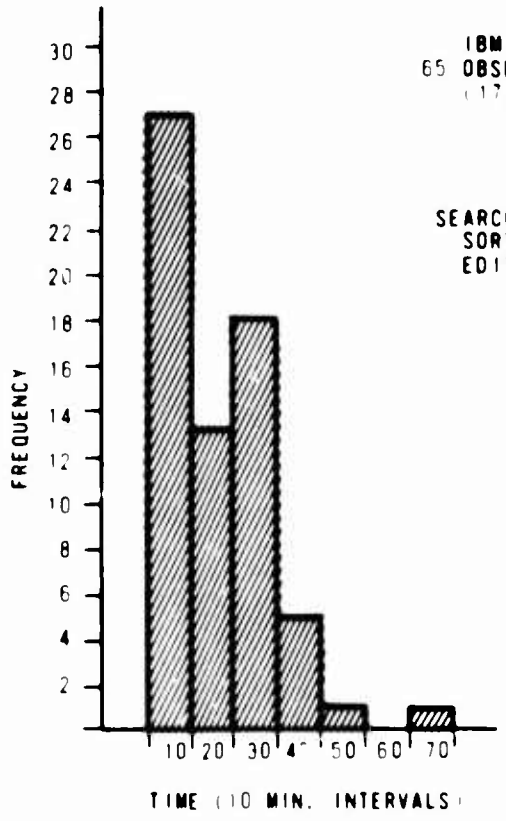
(a) time distribution tables and

(b) event time formulae.

An event may be expressed by either of these (or by combinations of both of these) methods. Figure 10 illustrates some distribution graphs made during







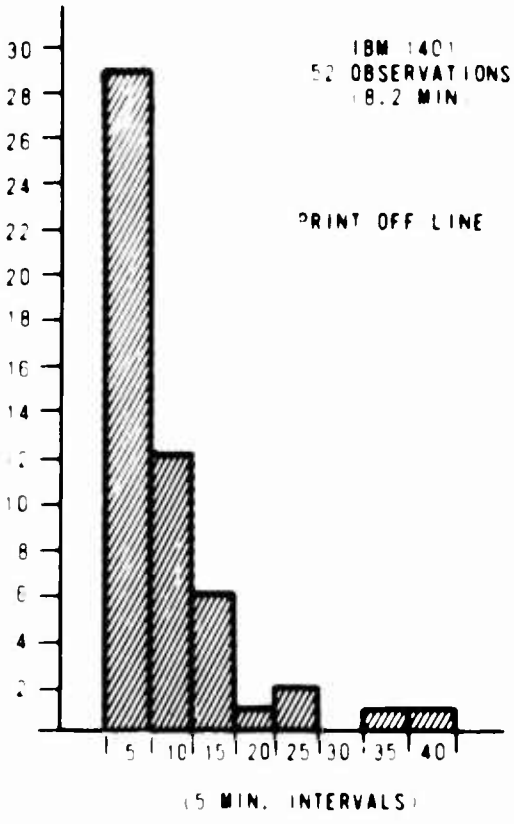


FIG. 10 MACHINE OPERATIONS

an operational test and evaluation of an ADP information storage and retrieval system. If these time distributions are representative of the time expenditures in the processing events; then for each time range, the ratio (frequency of occurrence)/(sample size) can be treated as the probability for the processing time interval to cover that event.

For example, the off-line printing operation depicted in Figure 9 would have a probability time distribution as:

Print Time (Min.)	Probability	Cumulative Probability
[5, 10)[1]	.558	.558
[10, 15)	.231	.789
[15, 20)	.115	.904
[20, 25)	.039	.943
[30, 35)	.019	.962
[35, 40)	.019	.981
[40, 45)	.019	1 000

Selection of random number R = .632 would select a print time of [10, 15); similarly, R = .953 represents a print time of [30, 35).

Time distribution tables can be accurately provided by sampling the processing operation of the event in the system to be simulated or by examining a system containing a similar processing event.

Event time formulae have been developed by considering how the variables (e.g., equipment, files, query, etc.) in the processing event interrelate with respect to time. Figure 11 gives an expansion of the general logic of the Event Sequence Generator and illustrates the various points where processing times are calculated or selected from time distribution tables. The following discussion essentially considers only the development of time formulae, further remarks concerning time distribution tables will be given in a later report.

---

[1] The notation [5, 10) represents the interval $5 \leq T < 10$.

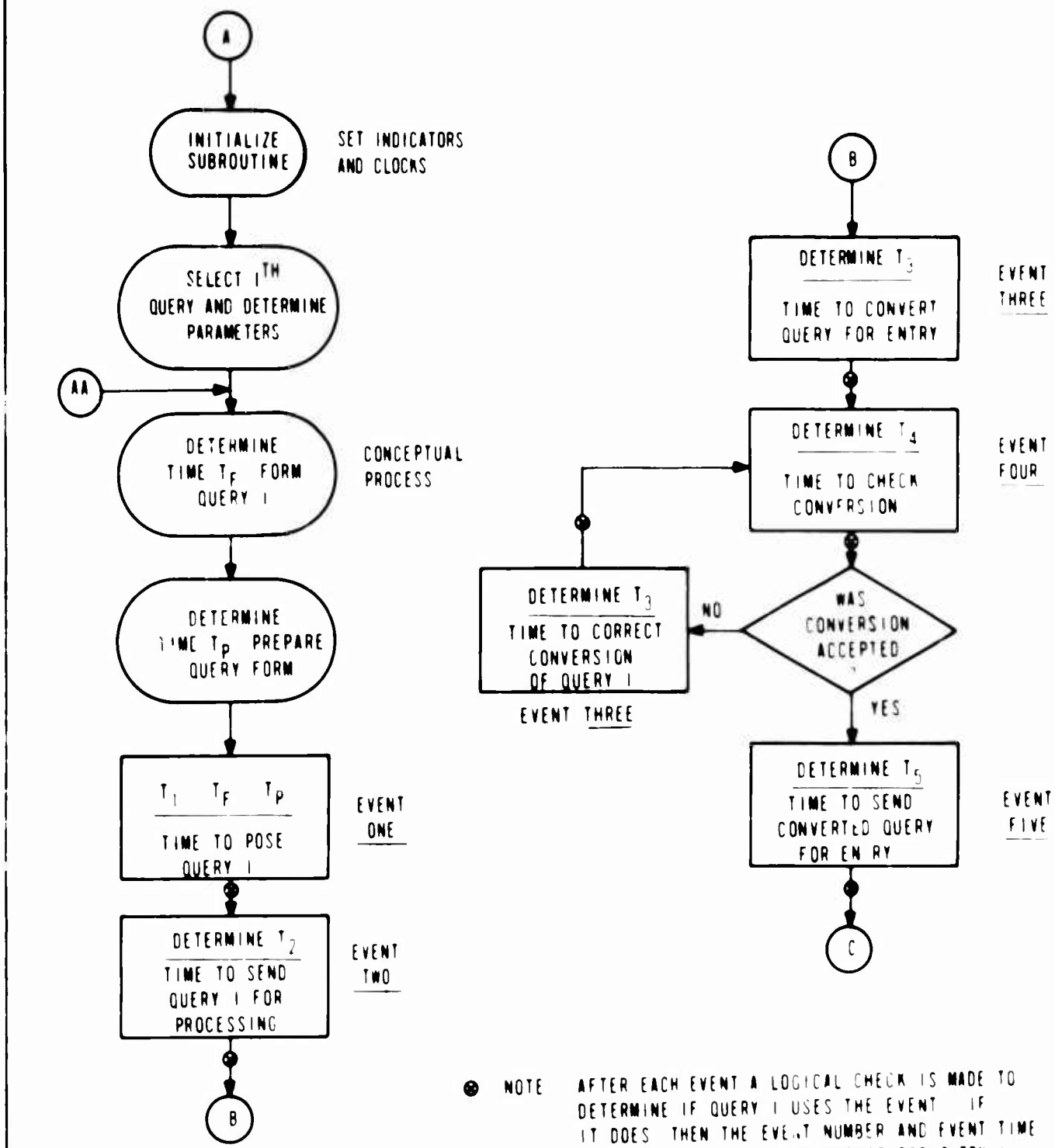


FIG 11 FLOW DIAGRAM OF EVENT SEQUENCE GENERATOR SUBROUTINE

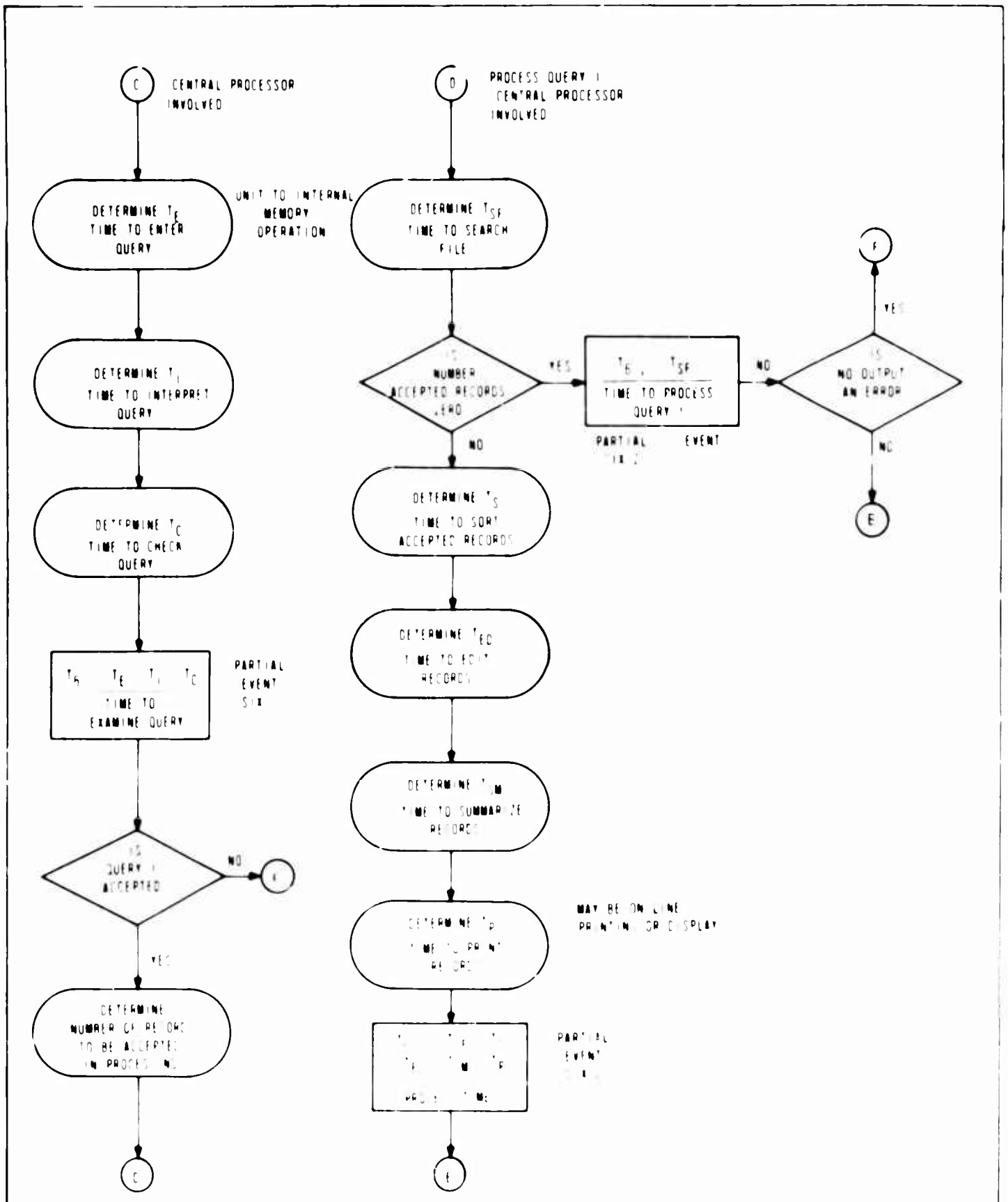


FIG 11 FLOW DIAGRAM OF EVENT SEQUENCE GENERATOR SUBROUTINE CONT D

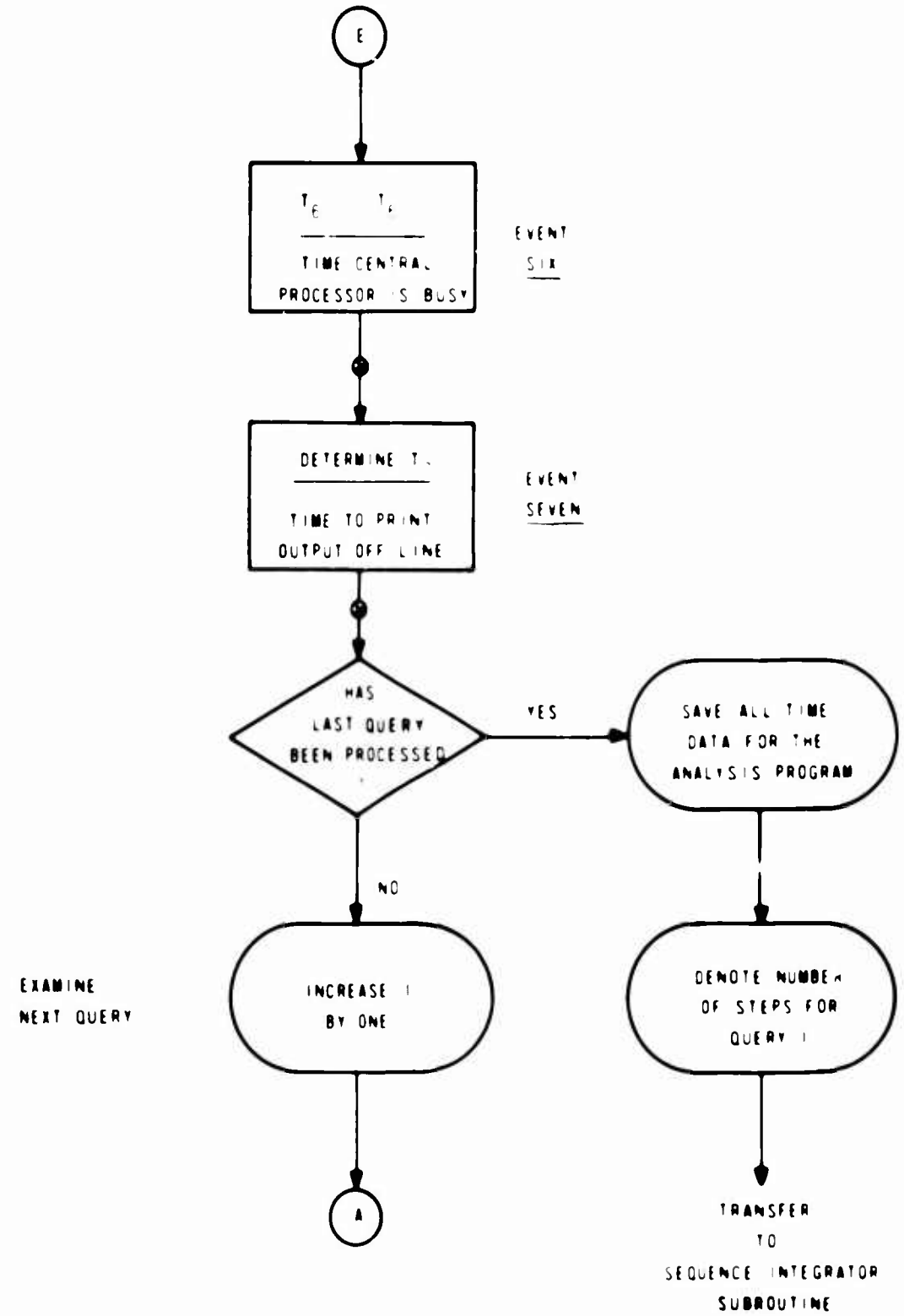


FIG. 11 FLOW DIAGRAM OF EVENT SEQUENCE GENERATOR SUBROUTINE (CONT'D)

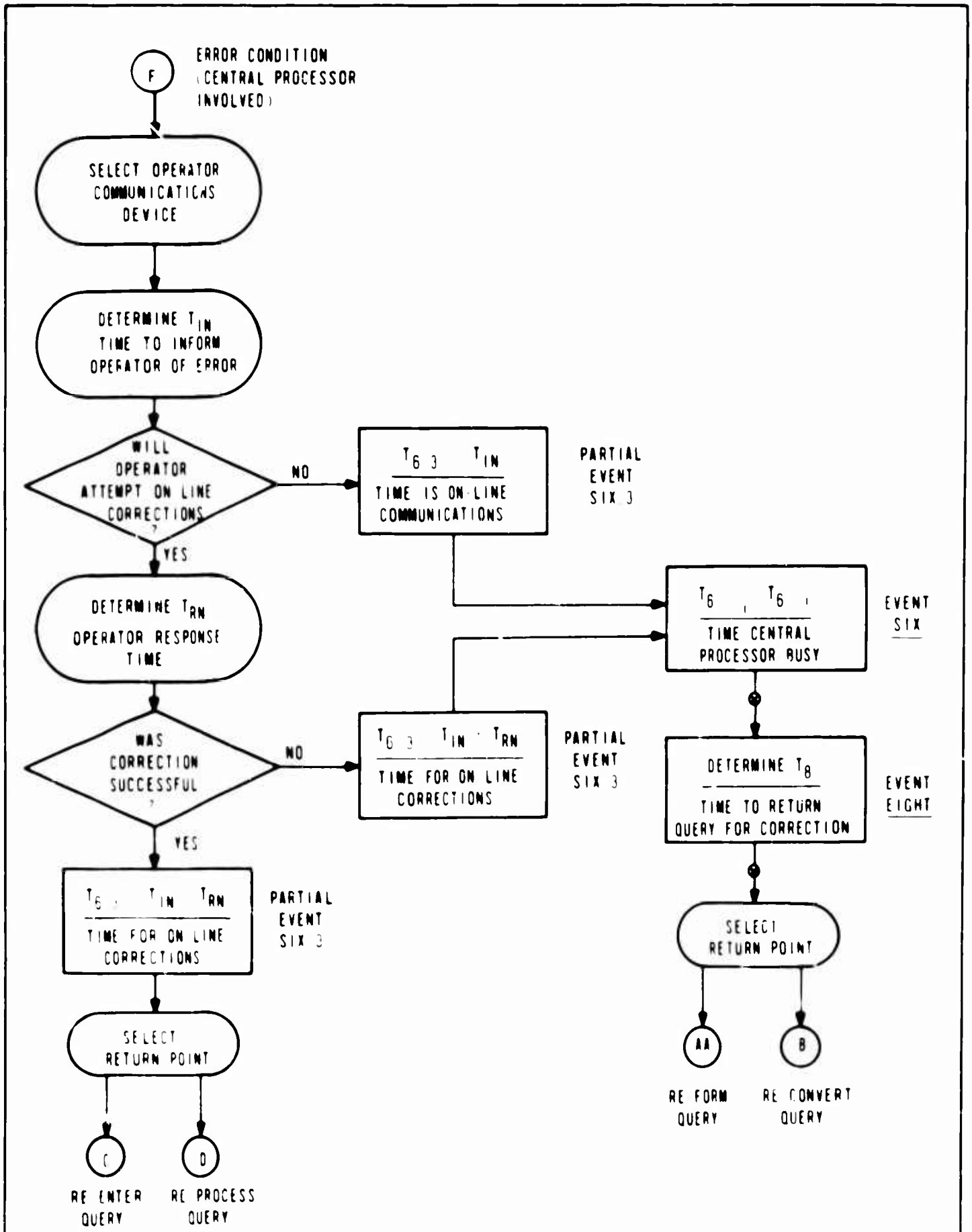


FIG 11 FLOW DIAGRAM OF EVENT SEQUENCE GENERATOR SUBROUTINE (CONT'D)

a. Query Parameters

There are several parameters that may be associated with the formal expression of a request and will influence subsequent processing time. For example

(1) query priority - may determine processing path and treatment in queue,

(2) number of items/characters - influences query preparation and entry time, may influence send (transmission) time,

(3) complexity of logic -- the nature and number of disjuncts and conjuncts may influence item comparison time,

(4) output statement -- determines if selected records are sorted, edited and/or summarized,

(request -- search statement may influence how much of the data base will be searched.

In order to effectively use time calculations in the simulation of a system or system concept, one must be able to categorize the requests to be placed against the system; identify the time influencing parameters of these queries and define how these parameters can be related to the equipment and general processing characteristics given in the time formulae. Hence, when the I^{th} query is processed in the simulation, these parameters can be used by the program to influence time calculations at each step in the retrieval process.

b. Time to Pose Query ($T_I = T_F + T_P$)

The time required to pose a query can be considered to be the sum of the times required to form the query (T_F) and to prepare a query form (T_P). Formulation of the query is a conceptual process and will only be represented by time distribution tables. Similarly, query form preparation is a strictly manual operation and can probably best be represented with time distribution tables. It should be noted that each facet of posing a query can be represented with several distribution tables corresponding to different question categories.

Moreover, the time for event one need not be separated into two distinct parts (T_F and T_P) but may be treated as one event time distribution.

c. Time to Send Query for Processing (T_2).

This event may be performed by (1) dispatching the query by courier or (2) transmitting the query over the telephone, teletype or some other communications device. The first method of communication can best be represented by time distribution tables. In the second methods, however, transmission time can be calculated as

$$T_2 = (\text{Set-up time}) + \left(\frac{\text{nr. transmitted characters in query}}{\text{effective transmission rate}} \right)$$

Set-up time is the time expended in preparing the data or the device for the sending operation. In general, set-up time is dependent upon the operator and the device and is not a function of the data parameters. Each processing function, having a man-machine interface, may have some initial operations which can be classified as set-up time. These operations will be designed in subsequent event time expressions as ST.

The number of transmitted characters is a query and device parameter, i.e., gives the number of characters in the query plus the number of communications control symbols such as teletype carriage return, linefeed and spacing symbols. The effective rate of the communications device is a man-machine characteristic and represents the lowest average characters per second (CPS) rate of the man and the machine. Thus, a 10 CPS device operated by a 4 CPS operator has an effective rate of 4 CPS.

d. Time to Convert Query (T_3)

Query conversion may be a machine function (e.g., optical scanning of the query form) or a man-machine function (e.g., card punching). Conversion time can be calculated as

$$T_3 = ST + \left(\frac{\text{nr. converted characters}}{\text{effective conversion rate}} \right) + \left(\frac{\text{nr. skipped characters}}{\text{effective skip rate}} \right)$$

Again, the effective conversion rate is the lowest average CPS rate of the operator (if applicable) and the device. In some equipment (e.g., a card punch) there may be a significant difference between the effective conversion rate and the skipping rate. For example, in some operations, it has been found that the effective punching rate with the IBM 026 printing card punch is 1.67 CPS, the maximum skip rate for this device is 80 CPS.

In the flow chart given in Figure 10, event three is shown to encompass both conversion and correction. This configuration has been selected because it is frequently the case that the correction function is a reconversion effort. Therefore, it is possible that a queue could form at a conversion point with a mixture of data requiring conversion or reconversion. The conversion rates, however, may differ; hence, two conversion time expressions may be required.

EXAMPLE

Consider a query conversion process where query items are punched into fielded positions within a card check. Assume query type X_1 requires seven 80-column cards with a total of 155 punched characters.[1] Suppose that within each card there are five programmed skip zones, then the following data represent conversion parameters:

7 X 80 =	560	possible characters
	155	punched characters
	405	skipped characters

Since the 35 skip zones require operator keying, these 35 key strokes can be added to the 155 punched characters to give 190 operator strokes at an effective rate of 1.67 CPS. The time required to convert query type X_1 can be calculated as.

[1] Columns individually skipped are counted as being punched since this operation requires an operator to depress a key in the same manner as punching a character.

$$T_3 = ST + \left(\frac{190}{1.67} \right) + \left(\frac{405}{80} \right) = 119.4 \text{ seconds}$$

or about 2 minutes plus the set-up time.

If the query conversion is not accepted, the conversion parameters may be modified and the query (or parts of the query) reconverted. This conversion effort may be accomplished at the same initial conversion rate (but under new parameters, e.g., 2 cards with 35 punched characters) or with a new rate. For example, an effective card correction rate utilizing card duplication could be 18 CPS (as opposed to 1.67 CPS).

e. Time to Check Conversion (T_4)

The time expended to check the accuracy of query conversion can be expressed as

$$T_4 = ST + \left(\frac{\text{nr. elements checked}}{\text{effective check rate}} \right) + \left(\frac{\text{nr. elements skipped}}{\text{effective skip rate}} \right).$$

Again, the effective rates are for a man and/or device component.

EXAMPLE:

Assume that a retrieval system uses a punched card deck for the query input medium and that this conversion effort is checked by visual scanning. Suppose that test experimentation reveals that the checking rate is .51 fields per second and is a function of the number of fields used, i.e., blank fields are ignored by the operator. Query type X_1, with a parameter of 32 fields, would require 62.7 seconds, i.e.,

$$T_4 = ST + \left(\frac{32}{.51} \right) + 0 = 62.7 \text{ seconds},$$

or a little over a minute plus set-up time.

f. Time to Send Query for Entry (T_5)

This function represents the transfer of the converted query to the computer room for input The time required for this operation can be best denoted with time distribution tables.

g. Time to Process Query (T_6)

Event six represents the complex functions involving the central processor. Imbedded within this event are ten distinctive operations, i.e.,

(1) enter query -- the process of placing the query into internal memory. This may represent a card-to-core operation through a card reader, an on-line request using a query console, etc.

(2) interpret query -- translating a natural or symbolic query language into machine language.

(3) check query -- program examination of query to check accuracy of input statement.

(4) search file - linear search of data base and logical comparison of file records with query.

(5) sort records -- arrangement of selected records into specific sequence.

(6) edit records -- preparation of selected records for output display.

(7) summarize records -- reduction of selected records for statistical presentation.

(8) print records -- on-line display.

(9) inform operator of error -- during processing, error conditions are analyzed by the program and the operator is notified by on-line error messages.

(10) on-line correction -- the operator attempts to correct the difficulty without releasing the central processor for another job.

It is not necessary that all operations are performed in the processing of a query. The path through query processing is determined in the Event Sequence Generator from the query parameters and the system configuration specified to the simulation.

Processing time (T_6) is the sum of the times required in each partial event as illustrated in Figure 10. Following is a brief discussion of the time expressions provided to represent computer processing time expenditure.

Examine Query -- Partial Event (6, 1)

The time required to examine the query can be expressed as the sum of the times required to set up the process, enter the query, interpret and check the query, i.e.,

$$T_{6,1} = ST + T_E + T_I + T_C.$$

Set-up time (ST) will be dependent upon the facility requirements to mount tapes and initialize the retrieval program. The other time expressions are functions of the query parameters and equipment rates, i.e.,

ENTRY TIME $$T_E = (\text{device start time}) + \left(\frac{\text{nr. accountable characters in query}}{\text{character rate of device}} \right).$$

Where device start time may represent initial communications with remote stations, the accountable characters may include blanks and control symbols in addition to the characters of the request.

INTERPRETATION TIME $$T_I = \left(\frac{\text{nr. items}}{\text{interpretation rate}} \right)$$

CHECK TIME $$T_C = \left(\frac{\text{nr. items}}{\text{checking rate}} \right)$$

EXAMPLE:

A remote query station communicates with a processing center. After the processing instructions have been given to the operator and he has initialized the system; the remote station is linked with the processor for direct query entry. Initial communications average 38.6 seconds and the setup time is observed to be 146 seconds. Query X_i contains 420 characters that have been placed on punched tape in a prior conversion effort and will be transmitted into the system at 20 cps. The data are transmitted in machine language; however, there are twenty items in the query that are checked by the program with a rate of 400 items per second. The time used in examining query X_i is

$$ST = 146.0$$

$$T_E = 38.6 + \left(\frac{420}{20}\right) = 59.6$$

$$T_I = 0.$$

$$T_C = \frac{20}{400} \quad \frac{}{T_{6,1}} = .05 = 205.65$$

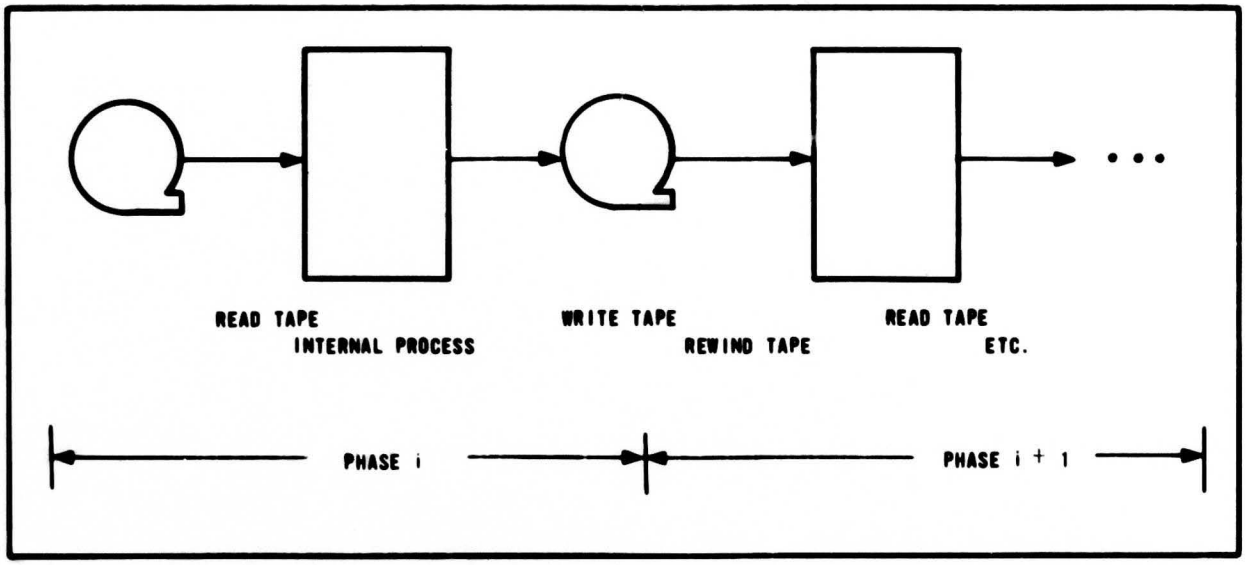
Process Query -- Partial Event (6, 2)

The time required to process the query, once the query has been accepted by the central processor, is the sum of the times required to search the file, sort, edit, summarize and print the selected records, i.e.,

$$T_{6,2} = T_{SF} + T_S + T_{ED} + T_{SM} + T_P.$$

These time expressions are dependent upon the query parameters, equipment rates, file structure and processing techniques.

In general, each of the first four distinct phases of processing within partial event (6, 2) can be depicted as



The time used in the i^{th} processing phase can be denoted by

$$T_i = ST + T_R + T_I + T_W + T_{RC} + T_{RW}$$

where

ST is operator setup time

T_R is the time required to read the file

T_I is the time required in internal processing

T_W is the time required to write the output records onto output tape

T_{RC} is recovery time, i.e., time used by the program to read or write past a tape redundancy stop.[1]

T_{RW} is tape rewind time for the output tape feeding the next processing phase.

---

[1] During large volume tape processing, tape read-write heads may collect dust that is recognized as a bit, and a parity error may occur. In many programs, various tape movement schemes are used in an attempt to shake off the dust and recover correct processing.

These different facets of processing may not apply to every phase of processing among different systems. Moreover, in some instances where they do apply, the time expenditures do not increase response time. Tape write time, for example, may be imbedded within tape read time. The determination of when and what facets of each processing phase contribute to the overall response time expression is an important part of system analysis for the simulation effort.

It is presently estimated that the time expenditures for operator setup and for read/write recovery can be best provided with time distribution tables. Tape read time, tape write time, tape rewind time and internal processing time may be expressed either by distribution tables or by time formulae. The following discussion presents time formulae for the different internal processing time expenditures and for the basic tape movement times. For convenience, processing time in each i^{th} phase will be expressed as

$$T_i = ST + (T_R + T_W + T_{RC} + T_{RW}) + T_I$$

$$= ST + (\text{Tape Time}) + (\text{Internal Processing Time}).$$

TAPE TIME:

READ TIME

$$T_R = (P + E)\left(\frac{L}{R}\right) + E\,(\text{Start + Stop Time})$$

where P is the number of blocks passed (only pertinent to File Search)

E the number of blocks examined

L the length of a block on tape

R the inch-per-second rate of the tape drive.

The length of a block is determined by the character density of the tape, the number and size of the records in the block and the interblock gap, i.e.,

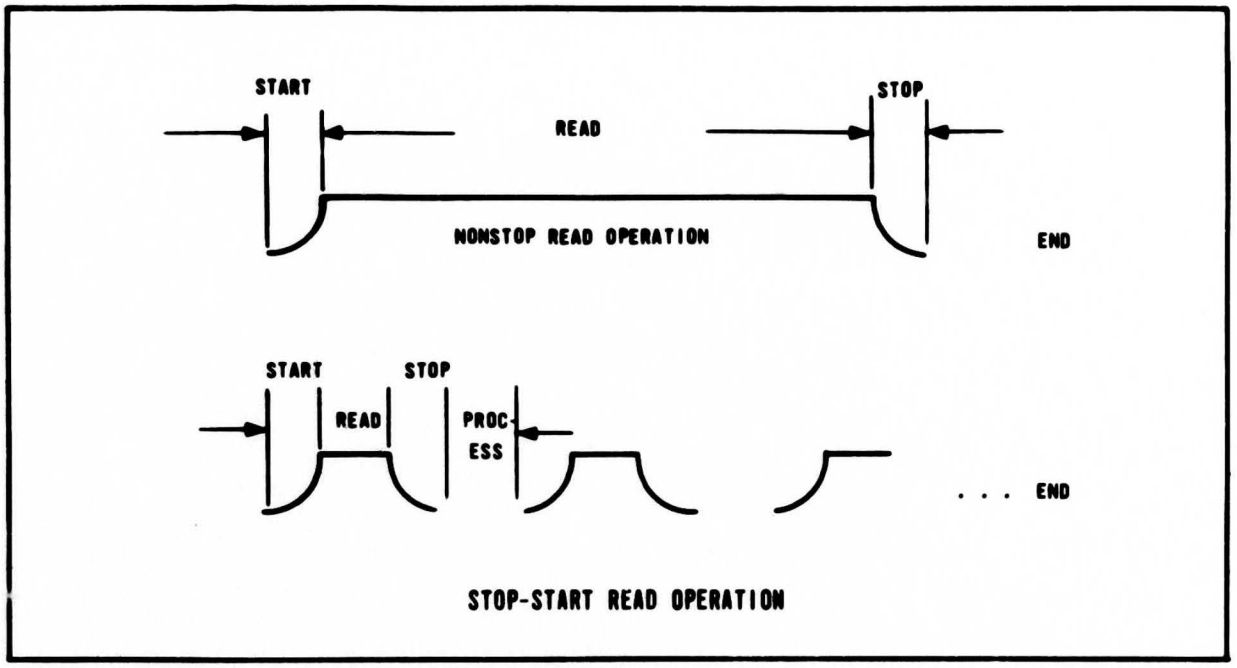
$$L = \text{gap} + \left(\frac{\text{nr. records}}{\text{block}}\right)\left(\frac{\text{nr. characters}}{\text{block}}\right)\left(\frac{1}{\text{character density}}\right).$$

In general, there are two distinct ways in which file reading may be pictured,[1] i.e.,

(1) Nonstop operation -- tape unit is kept busy and tape moves continuously. Processor gives command to read next record before tape slows down to stop.

(2) Stop-Start operation -- tape unit stops after each record is selected.

These two reading operations can be illustrated as



STOP-START READ OPERATION

[1] "Interaction of Hardware and Software Parameters in Tape Operations," W. B. Edwards, Jr. Proceedings ACM 20th National Conference/1965; pages 54-65.

In the read time expression, P represents the number of blocks passed in nonstop reading; E represents the number of blocks passed in a stop-start reading operation. Processing time, not imbedded in the start, stop and read operations, is not included in the read time expression; but is instead calculated in the internal processing time expressions.

The read time expression (T_R) provides for file reading to be

(1) a completely nonstop operation. For example, if there are N blocks on a tape; then by letting P = N-1 and E = 1 the read time expression becomes

$$T_R = N \left(\frac{L}{R}\right) + 1 \text{ (Start + Stop time)}$$

(2) a completely stop-start operation, i.e., let P = 0 and E = N, then

$$T_R = N \left[\left(\frac{L}{R}\right) + \text{(start + stop time)}\right]$$

(3) a mixture of reading operations. For example, a file may be ordered so that some general sections may be applicable to a request and all other file sections are inapplicable. The comparison logic may quickly reject inapplicable blocks and maintain continuous file reading. Applicable blocks, however, may require more extensive examination and the processor may not be able to complete the record comparisons before the tape unit stops.

EXAMPLE:

A tape file has a computer format with three records per block. The records are homogeneous, each containing 570 characters. The records are separated by 12 special symbols and the blocks are separated by a .75 inch interblock gap. Tape density is 556 characters/inch. Start time is 10.5 ms and stop time is 2.1 ms; tape unit has a read rate of 112.5 inches per second.

The effective length of a block can be calculated as

$$L = .75 + [(3)(570) + 24] \frac{1}{556 \text{ cpi}} = 3.85 \text{ inches.}$$

Consider a packed reel of tape 2400 feet long with approximately 15% of the date blocks potentially applicable to the search (the remaining 85% are rejected under continuous tape motion). The tape reel contains approximately 1,122 "examined" blocks and 6,358 "passed" blocks. The tape read time for this reel can be calculated as

$$T_R = (6358 + 1122)\left(\frac{3.85}{112.5}\right) + 1122\,(10.5 + 2.1) \times 10^{-3} \text{ seconds}$$

$$= 268.9 \text{ seconds (about 4.5 minutes).}$$

WRITE TIME

$$T_W = \frac{O}{R}\left[C + \frac{I \cdot d}{M}\right] + \frac{O}{M}[\text{start} + \text{stop time}]$$

where

- R is the recording rate of the device in characters per second
- O is the number of output records
- C is the number of characters to be written per record
- M is the blocking factor
- I is the length of the interblock gap in inches
- d is the tape density in characters per inch.

Tape writing time is essentially a function of the recording speed of the equipment and the volume of data to be recorded.

The number of interblock gaps are integral; however, non-integral values of $\frac{O}{M}$ should not significantly affect time calculations; hence, $\frac{O}{M}$ will be treated as a simple fraction in the simulation program.

EXAMPLE

200 records are to be written in blocks of four records. Each record contains 355 characters. If these records are written out on tape with a tape unit having the following characteristics

R = 15,000 characters per second

I = .75 inch interblock gap

d = 556 characters per inch

Start time = 7.5 ms

Stop time = 5.1 ms,

then the write time can be calculated as

$$T_W = \frac{1}{15,000} \times 200 \left[355 + \frac{1}{4}(.75 \times 556)\right] + \frac{200}{4}(7.5 + 5.1)\,10^{-3}$$

$$= 6.65 \text{ seconds}$$

RECOVERY TIME

Read/Write recovery time (T_{RC}) is a software parameter (i.e., the number of attempts to read or write and the techniques used to move the tape are programmed into the system). The number of redundancy stops encountered in processing is frequently a function of the amount of high speed tape movement; thus, different processing phases may have different expected recovery times. Presently, recovery times for the different processing phases within event 6 are to be provided from time distribution tables.

REWIND TIME

$$T_{RW} = \frac{X}{R} + t_c \quad \text{if } P \cdot L \geq X$$

$$T_{RW} = \frac{P \cdot L}{R} \quad \text{if } P \cdot L < X$$

where

X is the length of tape (in inches) examined for beginning tape load point

R is the rewind rate in inches per second over X

P is the number of blocks on the output tape

L is the effective length of a block on tape

t_c is the average high speed rewind time.

There are normally two rewind speeds for a tape unit, i.e., a high speed rewind that is in effect when the tape under the read/write heads is beyond some given distance X from the load point; and a slower speed for rewinding within the given distance. Since the high speed rewind accelerates somewhat proportionally to the volume of tape, the overall high speed rewind time is somewhat constant (t_c).

EXAMPLE:

A tape unit rewinds with a constant speed of 75 inches per second within 450 feet from the load point. Elsewhere, the high speed rewind averages 1.2 minutes for a reel of tape from 450 to 2400 feet in length. An output tape contains 200 records in one record blocks. Each block (including an interblock gap) is 1.77 inches. The rewind time can be calculated as

$$T_{RW} = \frac{200 \times 1.77}{75} = 4.72 \text{ seconds.}$$

INTERNAL TIME:

SEARCH FILE TIME $T_{SF} = ST + (\text{Tape Time}) + T_{CN}$

where T_{CN} is the time required in comparing the structure and content of a query with the contents of the file.

COMPARISON TIME

$$T_{CN} = (t \times 10^{-6})[(C_i \cdot I + C_r) R + (C_i \cdot I' + C'_r) R']$$

where

t is the processor cycle speed in microseconds

Ci is the average number of cycles used per tested item

Cr (Cr') is a number of cycles associated with the manipulation of each accepted (rejected) record

R (R') is the number of examined records accepted (rejected) in a a file search.

If part of the comparison time is imbedded within read/write time, the additive portion of the comparison time is the non-negative value for

$$T_{CN} - \left(\frac{\text{Overlapped time}}{(\text{block})}\right)\left(\frac{1}{(\text{records/block})}\right)(R + R').$$

The variables I, I', R and R' are parameters of each query under the simulation; C_i, C_r and C'_r are software characteristics and t is an equipment characteristic. A detailed study of this comparison time expression is given in Appendix A of this report and will not be repeated here. However, it should be noted that the software characteristics can be estimated from an analysis of the IR program or logic flow chart, and the query parameters can be derived from an analysis of the contents of the data base and the query structures considered. These estimates may not be precise, hence, error may be introduced into the calculation of T_{CN} (and other response times). Appendix A also contains some study notes on the effects of error in the engineering estimates of the processing functions.

EXAMPLE:

A central processor has a 12 microsecond cycle time speed in machine processing. In a pass against a full tape reel, 22,640 records are examined. It is estimated that under query X_i

R = 200 accepted records

R' = 22,240 rejected records

I = 2 (Average number of items tested)

I' = (exact number of items tested per rejected record)

Moreover, an engineering study of the processing concept yields the following table.

	cycles per accepted record	cycles per rejected record
"AND" QUERY	$18\,I + 34$ $I = N$	$18\,I' + 26$ $1 \leq I' \leq N$
"OR" QUERY	$18\,I + 32$ $1 \leq I \leq N$	$18\,I + 28$ $I' = N$

Where N is the number of items specified in the query. From this study, a comparison time formula is expressed as

$$T_{CN} = (12 \times 10^{-6})\,[(18 \cdot I + 33)\,R + (18 \cdot I' + 27)\,R'].$$

For all query classes in the IR systems, the comparison time used in the file search for query X_i can be calculated as

$$T_{CN} = (12 \times 10^{-6})\,[(18 \cdot 2 + 33)\,200 + (18\ 4 + 27)\,(22,240)] = 26.6 \text{ seconds.}$$

SORT TIME

T_S = ST + (Tape Time) + (Internal Sorting Time).

There now exists a large and expanding literature treating sorting methods and techniques. Among these works, the operating times of the different sorting programs have been estimated with respect to the parameters of the file, the number of keys in the sort and the speed and capacity of the central processor.

The 1963 May issue (volume 6 number 5) of the Communications of the ACM gives the papers presented at the ACM Sort Symposium and represents

a significant collection of recent works on sorting. Additionally, each computer manufacturer normally provides some expression of the sorting time required under different program/processor configurations. The problem confronting the present effort in developing the response time simulation program is to provide a method of adequately expressing these various time formulae for the different sorting concepts. At the moment, this problem has not been solved and is a subject of the current work effort.

EXAMPLES

The following examples illustrate some of the work now available in calculating sorting time. Gotlieb gives the following sorting time expression[1]

$$T = \max (T_{P1}, T_{T1}) + \max (T_{P2}, T_{T2}) + T_{T3}$$

where:

T_{P1} is the process time for the internal sort. It depends on the cycle time of the computer, the method chosen for internal sorting, the storage available, the record size and the number of words in the key. The first three of these depend on the computer and program, and the others are parameters of the file.

T_{T1} is the time for reading the file, initially arranged in blocks B1, and writing in blocks of a size determined by G, the group of records produced by the internal sort.

T_{P2} is the process time for sorting the file into individually sequenced tapes. It depends on the cycle time and on the merging process, which in turn depends on the number of tape units available.

T_{T2} is the tape time for the merge required to produce individually sequenced tapes. It is proportional to the number of passes.

T_{T3} is the tape time for collating the individually sorted tapes, this time which is proportional to the number of tape reads is greater than the process time of stage 3.

The article also gives expressions for the number of passes required in merging sorting strings; e.g., if 2p tape units are available, the number of passes in a p-way merge are $[\log_p N]$ where N is the initial number

---

[1] C. C. Gotlieb, "Sorting on Computers," Communications of the ACM; May, 1963, pp. 194-201.

of strings and the bracket notation ([ ]) indicates the smallest integer equal to or greater than $\log_p N$. Thus, a 2-way merge of 200 records grouped into 20 strings would require 5 passes if 4 tape units are available. If the time required to compare keys is less than that needed to read the records, the sorting speed is tape limited, i.e., determined by the read/write speed of the processor and the number of passes required.

In the same issue of the Communications of the ACM, Hall gives time formulae for calculating the interval sorting time for sixteen different sorting methods.[1] For example, a p-way merge[2] would have a calculated internal sorting time of

$$T_{pi} = C_t (p - 1) N [\log_p N] + T_t [\log_p N]$$

where:

C_t is the time required to compare two sort keys

T_t is the time required to transfer an item from one location in memory to another

N is the number of items in the file.

Again, the bracket notation represents the least integer greater than or equal to $\log_p N$.

EDIT TIME

$$T_{ED} = ST + (\text{Tape Time}) + T_{ed}$$

$$T_{ed} = (t_1 I_1 + t_2 I_2) A$$

where:

t_1 is the time required to test an item

I_1 is the number of items tested

---

[1] Michael H. Hall, "A Method of Comparing the Time Requirements of Sorting Methods," pp. 259-263.

[2] Hall's notation has been slightly modified to be compatible with that given by Gotlieb.

t_2 is the time required to transfer an item from one location in memory to another

I_2 the number of items transferred per record

A the number of accepted records edited.

This expression of the internal processing time required to edit a file considers that the basic editing operations, i.e., add data (such as special symbols, spaces, etc.), delete data and rearrange data, can be treated as a simple transfer function. File records are read into internal storage cells and then transferred, under program control, into storage cells that are the image of the output requirements. The data are then read out of these latter storage positions onto the output tape. Figure 12 illustrates this transformation.

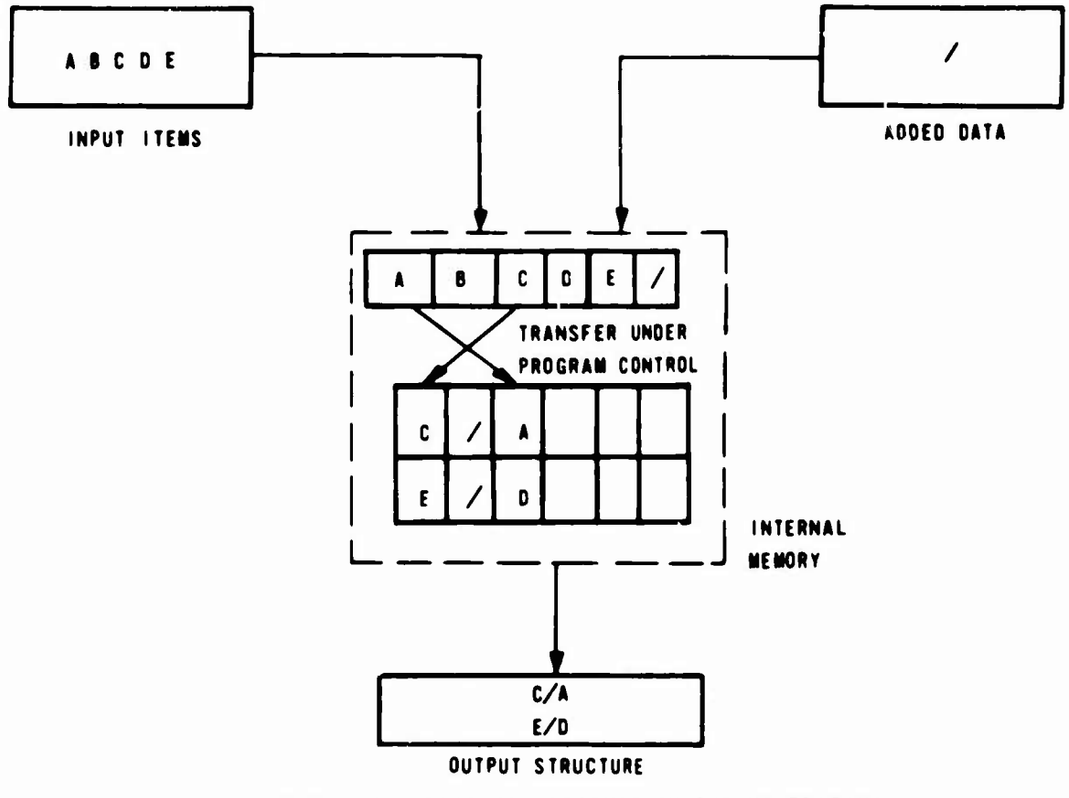


FIG. 12 DATA TRANSFORMATION DURING EDITING

Editing time may be imbedded within the tape read/write time or some other function such as sorting or summary. For example, in the initial pass of a sort program, the selected file records may be read in, edited, compared and distributed among several tape units for subsequent sorting operations. The additive editing time is the internal editing time minus the overlapped time in processing.

EXAMPLE:

A file has been searched and 200 records have been selected and are to be edited. The following engineering data have been obtained and are applicable to the calculation of internal processing time for the editing operation.

> Each record contains 71 items. 10 items are to be added and 25 items are to be deleted. The cycle time of the processor is 12 microseconds. It is estimated that 18 cycles are required to test an item and 2 cycles are required to transfer an item. Hence,
>
> $$t_1 = 18\,(12 \times 10^{-6}) = 216 \times 10^{-6} \text{ seconds}$$
> $$t_2 = 2\,(12 \times 10^{-6}) = 24 \times 10^{-6} \text{ seconds.}$$

In the example edit program, both the input and the added items are tested; thus,

$$I_1 = 71 + 10 = 81 \text{ items.}$$

The number of items transferred is the total number of items minus those deleted; thus,

$$I_2 = 81 - 25 = 56 \text{ items.}$$

Therefore, the internal processing time is

$$T_{ed} = [(216 \times 10^{-6})\,81 + (24 \times 10^{-6})\,56]\,200 = 3.77 \text{ seconds.}$$

SUMMARY TIME

$$T_{SM} = ST + (\text{Tape Time}) + T_{sm}$$

$$T_{sm} = t(C_c A + C_t S)$$

where

t = cycle time of the central processor

C_c is the number of cycles required to check each accepted record

A is the number of accepted records

C_t is the number of cycles required to transfer the summarized data

S is the number of distinct summary records derived from the processing operation.

In general, there are two types of data summary, i.e.,

(1) A simple tabulation of "like" subsets within a sorted series of items.

(2) A statistical analysis of a given set of data.

This report is concerned with only the first of these two summary operations. Figure 13 gives a simple flow diagram exemplifying this summary operation. When the selected data file has been sorted under the summarization criteria, data summary can be treated as a simple comparison and counting operation. Like records (or record subsets) are counted; the occurrence of a "different" record (or subset) starts a new counting operation. Output is a group of distinct items representing distinct subsets within the file. The count gives the number of records having these properties.

EXAMPLE:

An IR system has a 12 microsecond central processor and a summary logic such as shown in Figure 13. The number of cycles required to check each accepted record is 64; the number required to transfer distinctive records is 76. It is estimated that query X_1 will select 200 records, having 25 distinctive categories. The time used in the internal process of data summarization is

$$T_{sm} = (12 \times 10^{-6})(64 \times 200 + 76 \times 25) = .18 \text{ seconds.}$$

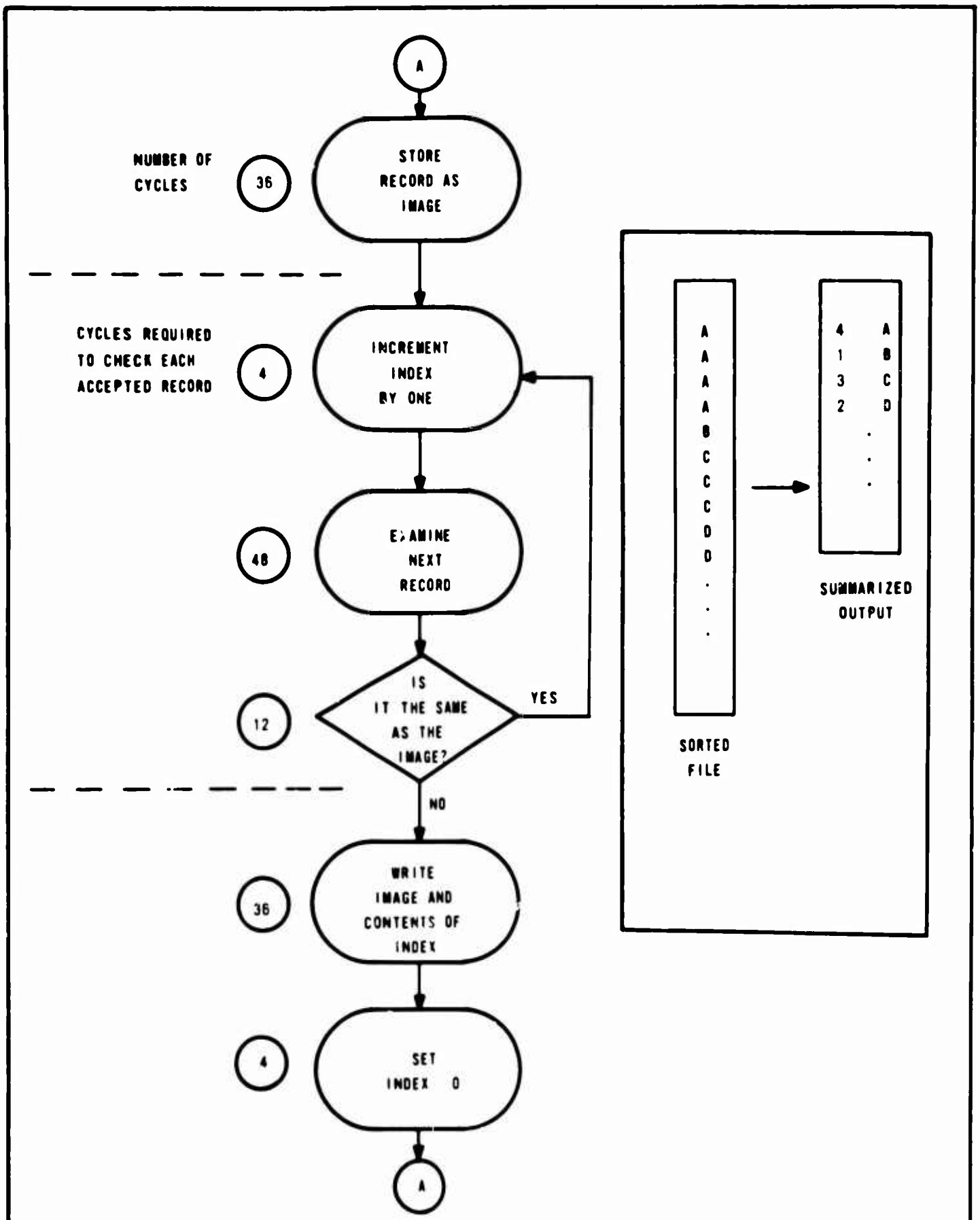


FIG. 13 SIMPLE FLOW DIAGRAM OF SUMMARY OPERATION

ON-LINE PRINT TIME

$$T_p = ST + \frac{C \cdot O}{R \cdot L}$$

where.

R is the lines per second output rate

L is the effective character length of a line

C is the number of characters to be displayed per record

O is the number of output records.

Tape time and rewinding are not included in this expression since on-line displays are usually bound by the speed of the display device and tape rewinding is not necessary to subsequent operations.

EXAMPLE:

200 records are to be printed with a ten lines per second on-line printer. The line width is 120 characters, however, it is estimated that 20 characters per line are used for spacing, thus reducing the effective length of the line to 100 characters. Each output record contains 355 characters to be printed. The on-line print time can be calculated as

$$T_p = ST + \frac{355 \times 200}{10 \times 100} = 71 \text{ seconds plus set-up time.}$$

Error Condition - Partial Event (6, 3)

Errors encountered during machine processing may delay the retrieval effort and tie up the central processor. Partial event (6, 3) provides a man-machine interface whereby (1) the central processor notifies the operator that some error has been encountered and (2) the operator may attempt on-line corrective action; thus

$$T_{6,3} = T_{in} + T_{rn}$$

the sum of the times required to inform the operator and the time used by the operator in response to the error condition. It is presently anticipated that these

time expenditures will be obtained from time distribution tables and will not be calculated.

Within partial event (6, 3), there are two decision points, i.e.,

(1) Will the operator attempt on-line correction or will he release the central processor and return the query for correction?

(2) Will the on-line corrective action be successful or will the operator have to terminate processing and return the query for corrective action?

These two decision points will be passed with the aid of a random number generator and the expected probability distributions provided by the investigating engineer.

h. Time to Print Output Off-Line (T_7)

The expression of time expenditure for off-line display is precisely the same as that for on-line display, i.e.,

$$T_7 = \frac{C \cdot O}{R \cdot L}$$

where

R is the lines per second output rate

L is the effective character length of a line

C is the number of characters to be displayed per record

O is the number of output records.

The difference between print time (T_p) and off-line print time (T_7) is that T_7 is not charged to the central processor, hence, the computer is free to process other queries while output is being printed.

i. Time to Return Query for Correction (T_8)

If the query failed to process correctly, it may be returned to be re-formed or reconverted. In either case, the converted query may be physically returned or there may exist some sort of communications exchange to determine the trouble. Event 8 provides for these delay times. Within event 8, the return points

AA -- reform query and

B -- reconvert query

are selected with a random number generator and an expected probability distribution.

B. SEQUENCE INTEGRATOR

The Sequence Integrator essentially operates upon two sets of data, i.e.,

(1) output from the Event Sequence Generator giving the sequence of events and processing times for each query of a question,

(2) input from the investigating engineer giving the number and arrangement of equipment and personnel available to each processing event.

In one sense, the Sequence Integrator performs the role of a scheduling director analyzing the data flow in the retrieval process and assigning work units to available equipment and personnel. Presently, the selection of data units from the queues is based on a first-come-first-serve policy with available equipment assigned from "left-to-right." Both the queue unloading stategy and the rule for assigning work to available service units may have several optional variations in future versions of the Sequence Integrator. These two aspects of work load and equipment use can influence the nature of the processing delays encountered in the retrieval effort; and they can influence equipment/personnel idle time.

1. Queue Unloading Strategy

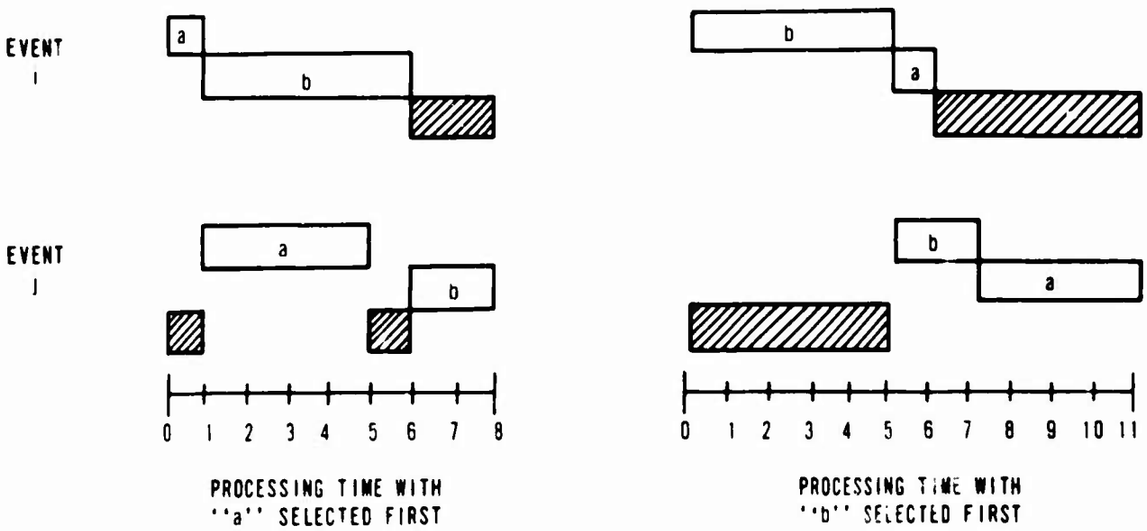
When two or more data units are waiting at the same event for servicing, it is often possible to expedite processing by selecting a data unit under some other criterion than first-come-first-serve. For example, selection of the data unit with the least expected servicing time may both reduce overall response time and equipment/personnel idle time. Figure 14 illustrates this point. In this example, data units "a" and "b" are in queue for event "i" and, when serviced, will proceed to event "j." Events "i" and "j" each have only one service element; hence can process only one data unit at a time. The difference in data selection causes an overall response time difference of 3 time increments.

Another criterion of data unit selection is to process the data units in a sequence such as to minimize computer idle time or to maximize utility of the central processor. For example, if in Figure 13, event j = 6 (i.e., the central processor is in use), then selecting "a" first in event "i" would give the computer two one-unit blocks of idle time. Selecting "b" first, would give the computer one five-unit block of idle time. Now, if a block of four units of computer time can be utilized for some other useful purpose, then selecting "b" first (and using the idle block for this other processing effort) would give the computer only one unit of idle time.

While different queue unloading strategies can produce different effects in processing, not all of these strategies are likely to be easily applied in the real world. Casual comparison of two data units may not effectively reveal which element will process faster in a given event, or which processing sequence will better contribute to effective utilization of the central processor. The introduction of sophisticated unloading strategies into the simulation may not reflect real world systems; however, if these strategies were available in the Sequence Integrator, design engineers could examine the differences these alternate unloading schemes cause in processing. Significant differences could point to a need to develop scheduling aids to expedite data flow within the processing center.

SEQUENCE OF EVENTS	QUERY a		QUERY b	
	EVENT	TIME	EVENT	TIME
K			i	5
K+1	i	1	j	2
K+2	j	4		

QUERIES "a" AND "b" ARE IN QUEUE FOR EVENT "i"



POTENTIAL EQUIPMENT PERSONNEL IDLE TIME

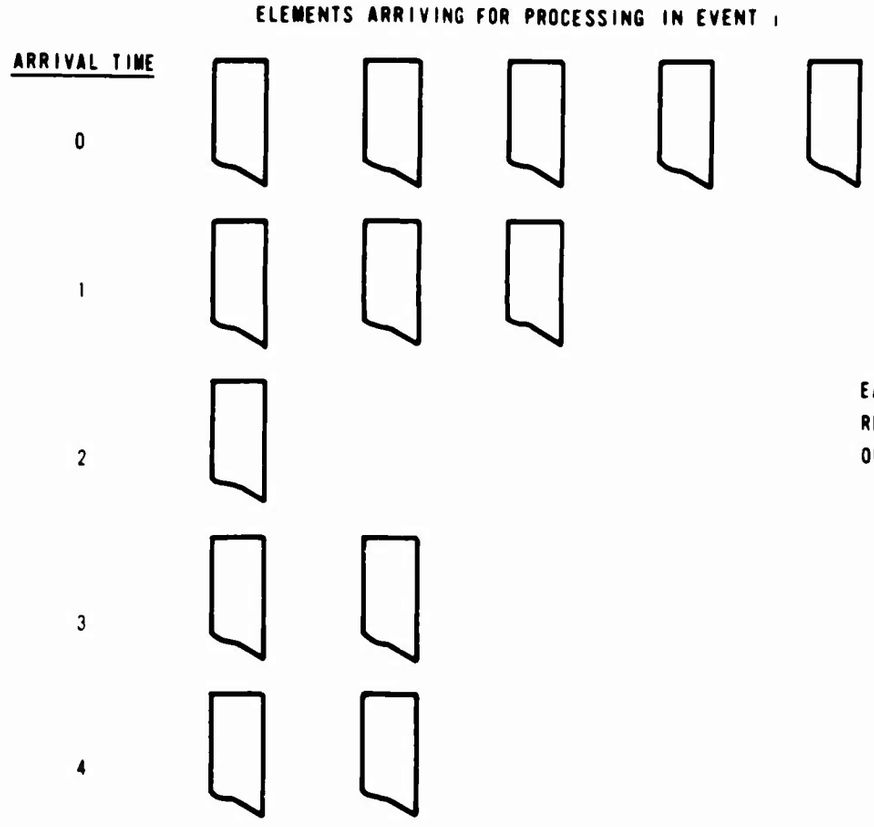
FIG. 14 COMPARISON OF PROCESSING TIME WITH DIFFERENT SELECTIONS FROM QUEUE

2. Service Unit Assignment

The assignment of available service units from "left-to-right" has several useful features in system simulation. The equipment needs of a system can be examined, for example, by initially providing an excessive number of service units for each event and placing low, normal and high processing loads against the system. Since the number of units available at each event will be more than adequate, there will be no processing delay caused by data units waiting for servicing. Because the service units are assigned in a given sequence, the program will assign work to the minimum number of units necessary for no-delay or minimum delay processing. Figure 15 illustrates how such an equipment study might look. In the first simulation run, the investigating engineer assigns six service units to event "i" and places some expected load factor on the system. The simulation output shows the operating time for each service unit and the total time data were delayed in waiting for service. In successive runs, the engineer reduces the number of service units while maintaining the same processing load. This will change the delay time in queue. An analysis of the cost per service unit, weighed against the effects of processing delays can aid in establishing an effective operating design.

Another use of this component reduction technique is to examine the potential effect of down-time. For example, if Figure 15 represented the processing capability of a facility over some time period T, then the loss of one service unit over time T would cause a work load shift and an increase in delay time. This type of study can be useful in system reliability analysis.

Work assignments, in the real world, do not strictly follow this left-to-right rule. An attempt is frequently made to distribute work uniformly among the available equipment and personnel. The Sequence Integrator should have a uniform work load assignment rule as an optional service unit assignment strategy, particularly if the simulation program is to cover situations where personnel share time in several operations, not all of which are connected with information retrieval.



	SERVICE UNITS FOR EVENT i						TOTAL TIME IN QUEUE
	1	2	3	4	5	6	
FIRST RUN	X	X	X	X	X	X	0
(OPERATING TIME)	5	4	2	1	1	0	
SECOND RUN	X	X	X	X			1
	5	4	2	2			
THIRD RUN	X	X	X				4
	5	5	3				
FOURTH RUN	X	X					17
	7	6					
FIFTH RUN	X						59
	13						

X INDICATES NUMBER OF SERVICE UNITS FOR EVENT i

FIG. 15 HYPOTHETICAL EXAMPLE OF EQUIPMENT STUDY

Reverse (Page 68) Blank

III. SUMMARY AND EXAMPLE

> "This view of a model as a design tool is one of the several roles which it can play in the development of a theory for this field [information systems]. In this usage, when the model is accepted as an adequate picture of a problem, the systems designer can specify the parameters of the problem in task-oriented terms and then ask...a computer to develop the consequences as prescribed by the model."[1]

The system engineer presently lacks sufficient tools to efficiently design, modify or evaluate complex information systems. This research into the simulation of information retrieval response time is to provide basic data necessary to the development of a performance measurement technique. A simple extension of the response time studies will provide a capability to estimate operating costs in the simulation program. Both response time and operating costs are necessary system evaluation criteria.

User satisfaction or acceptance of the IR system is another necessary evaluation criterion. Since this is a qualitative measure of system performance, it is not directly included within this research study. There are points, however, within the model outline that enable an investigating engineer to adjust and regulate the sensitivity of the system simulation. These controls can provide some insight into the cost of user dissatisfaction.

The ability to adjust and tailor the model outline to reflect specific computer IR systems (or system concepts) extends the potential utility of this tool. Presently, this outline does not cover systems having time-sharing or dynamic user/system interaction; but, instead emphasizes the service type computer systems employing linear file search.

The most important aspects of system definition within the general model framework are.

(1) Identification of the basic functions performed in the retrieval system.

---

[1] Becker and Hayes, Information Storage and Retrieval Tools, Elements, Theories; John Wiley & Sons, Inc., New York; p. 330.

(2) Classification of the queries posed against the system.

(3) Estimation of (a) the time required to process data through an event or (b) the basic parameters affecting processing time.

(4) Determination of the probable data flow through decision points within the retrieval operation.

During operation of the Event Sequence Generator, the events, operating time and processing sequence of a set of queries are determined. This determination is governed by the query parameters and the equipment characteristics fed into the simulation program by the investigating engineer. The parameters, in turn, can be derived or estimated from systems analysis, test experience or speculative studies of conceptual components. Output from the Event Sequence Generator is fed into the Sequence Integrator, which schedules the data flow through the IR system. Data units are taken from queue on a first-come-first-serve basis and are assigned to available service units from left to right. The Sequence Integrator keeps track of delay time in queue and equipment/personnel idle time during the retrieval process.

This work essentially completes the basic computer based model design. The next step to be taken will be the design of a model representing Information Retrieval systems which store their information in media such as, say, hard copy or microfilm. Libraries are an example of the systems to be considered. The equipment (such as card cabinets, shelf volume, etc.) and procedures (events) associated with such systems will be examined in detail -- this information will be incorporated into what will be called the Manual Model.

Following the design of the Manual Model, work will be started on the G-Model. This will be a combination of the manual and computer-based models and will represent a General Information Retrieval system. This model will be tested by using data on an existing Naval Intelligence System -- the IOIS.

Future efforts of this project can be directed toward research into several areas of information retrieval systems. The value of information retrieval systems which use time-sharing or multiprocessing computer systems is one such area. Also, total system operations which include input preparation and storage could be examined. Alternatively the operations involved in automatic

data processing (ADP) centers could be examined. In this case, the research would be directed toward job shop problems such as scheduling (to find out how to minimize turnaround time) or personnel requirements (to find out how to maintain a desired utilization level).

EXAMPLE:

Figure 16 gives a set of hypothetical data representing input into the Sequence Integrator for the response time analysis of a question posed to the simulated IR system. In this example, the system consultant prepares three queries to select data pertinent to the question. Queries 2 and 3 process without difficulty, however, query 1 is rejected by the central processor and is returned for partial reconversion. Figure 17 illustrates the processing of the three queries through the system and gives the output listing from the data analysis program. In this example, the time required to complete the retrieval effort for the question is 3400 seconds (56.6 minutes). Now assume that the investigating engineer analyzes the system configuration as follows.

(1) The "courier" function uses three people to deliver data (i.e., one in each event 2, 5 and 8) where two might do the job.

(2) One keypunch service unit should be sufficient for the query conversion effort.

In order to test the effect of system modification, based on this analysis, the engineer redefines the available service units as

Case 2

Event	Service Unit		Description	Number	
	1	2		P	E
1	X		System Consultant	1	
2	X	X	Courier	2	
3	X		Keypunch and Operator	1	1
6	X		Central Processor	2	1
7	X	X	Off-Line Printer		2
			TOTAL	6	4

SEQUENCE OF EVENTS	QUERY 1		QUERY 2		QUERY 3	
	EVENT	TIME	EVENT	TIME	EVENT	TIME
1	1	300	1	100	1	100
2	2	200	2	200	2	200
3	3	300	3	300	3	200
4	5	200	5	200	5	200
5	6	200	6	800	6	400
6	8	200	7	800	7	700
7	3	100				
8	5	200				
9	6	500				
10	7	500				

OUTPUT FROM EVENT SEQUENCE GENERATOR

ALL TIMES ARE GIVEN IN SECONDS

EVENT	SERVICE UNIT			DESCRIPTION	NUMBER	
	1	2	...		PERS.	EQUIP.
1	X			SYSTEM CONSULTANT	1	
2	X			COURIER	1	
3	X	X		KEYPUNCH AND OPERATOR	2	2
5	X			COURIER	1	
6	X			CENTRAL PROCESSOR	2	1
7	X	X		OFF LINE PRINTER		2
8	X			COURIER	1	
				TOTAL	8	5

AVAILABLE SERVICE UNITS

FIG. 16 EXAMPLE DATA FOR SIMULATION OF RESPONSE TIME FOR ONE QUESTION (CASE 1)

QUERY	REQD PROCESSING TIME	COMPLETION TIME
1	2700 00	3400 00
2	2300 00	2800 00
3	1800 00	3100 00

QUERY	END TIME
1	300 00
2	400 00
1	500 00
3	500 00
2	700 00
1	800 00
3	900 00
2	1000 00
1	1000 00
3	1100 00
2	1200 00
1	1200 00
3	1400 00
1	1400 00
2	2000 00
1	1500 00
3	2400 00
1	1700 00
1	2900 00
2	2800 00
3	3100 00
1	3400 00

EVENT	USE OF SERVICE UNIT 1	2
1	500 00	
2	500 00	
3	600 00	300 00
4	0 00	
5	800 00	
6	1900 00	
7	1300 00	700 00
8	200 00	

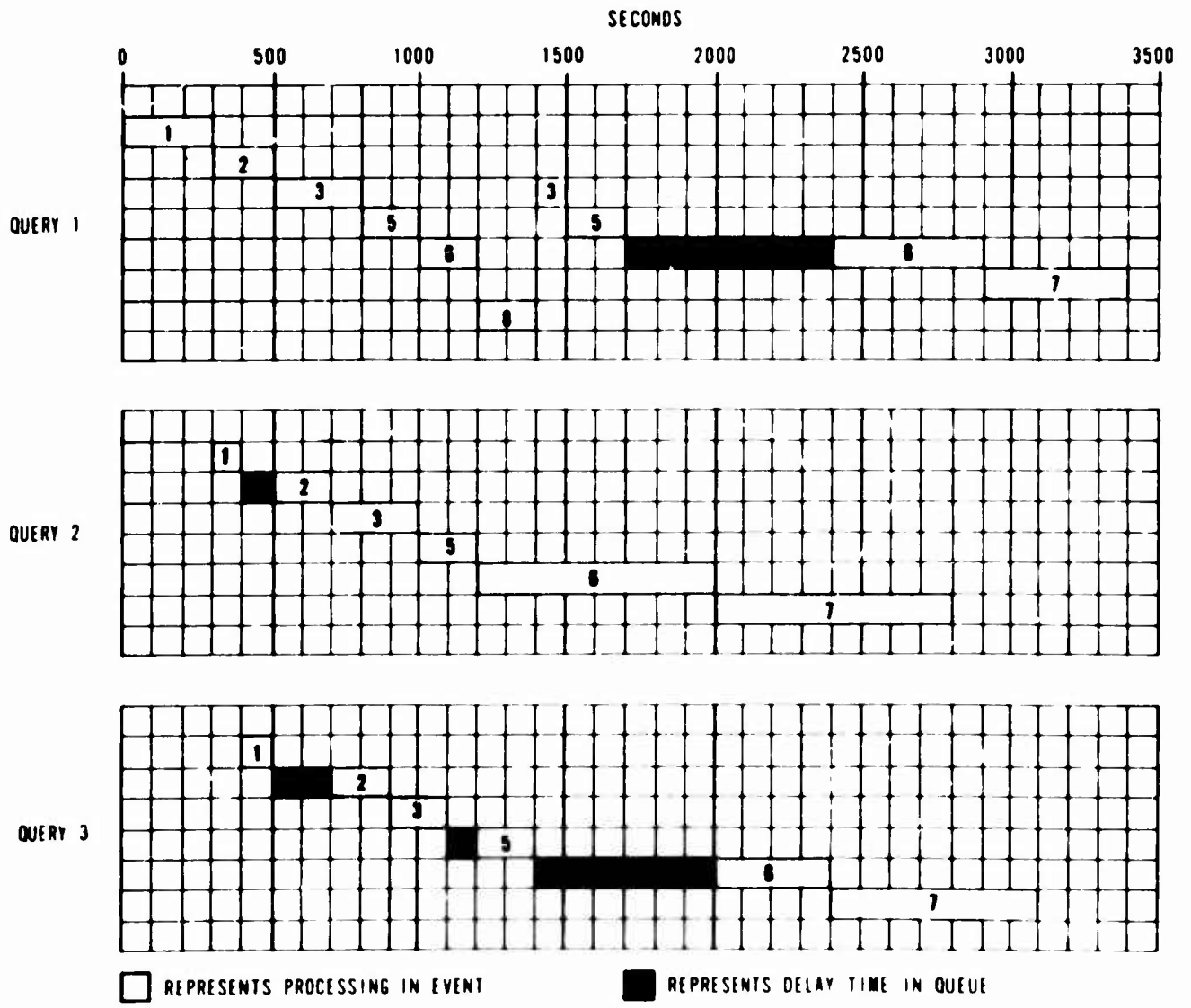


FIG. 17 INTEGRATED SEQUENCE OF EVENTS (CASE 1)

Within the Event Sequence Generator, events 5 and 8 are relabeled EVENT 2; hence, the two couriers perform the functions of transporting query i for

(1) processing,

(2) entry, and

(3) correction.

Figure 18 illustrates the reprocessing of the original three queries under the modified configuration. In the second processing effort, the overall delay time increased by 100 seconds with a reduction of two people and one equipment component from the processing system. Continued simulation, over a large representative question sample, could aid the engineer in determining which of these (or other) alternatives gives the best system capability within limits imposed by response time and cost requirements.

QUERY	REQD PROCESSING TIME	COMPLETION TIME
1	2700 00	3500 00
2	2300 00	2900 00
3	1800 00	3200 00

	USE OF SERVICE UNIT	
EVENT	1	2
1	500 00	
2	1200 00	300 00
3	900 00	
4	0 00	
5	0 00	
6	1900 00	
7	1300 00	700 00

QUERY	END TIME
1	300 00
2	400 00
3	500 00
1	500 00
2	600 00
1	800 00
2	1100 00
3	700 00
3	1300 00
1	1000 00
1	1200 00
2	1300 00
1	1400 00
2	2100 00
3	1500 00
1	1500 00
1	1700 00
3	2500 00
1	3000 00
2	2900 00
3	3200 00
1	3500 00

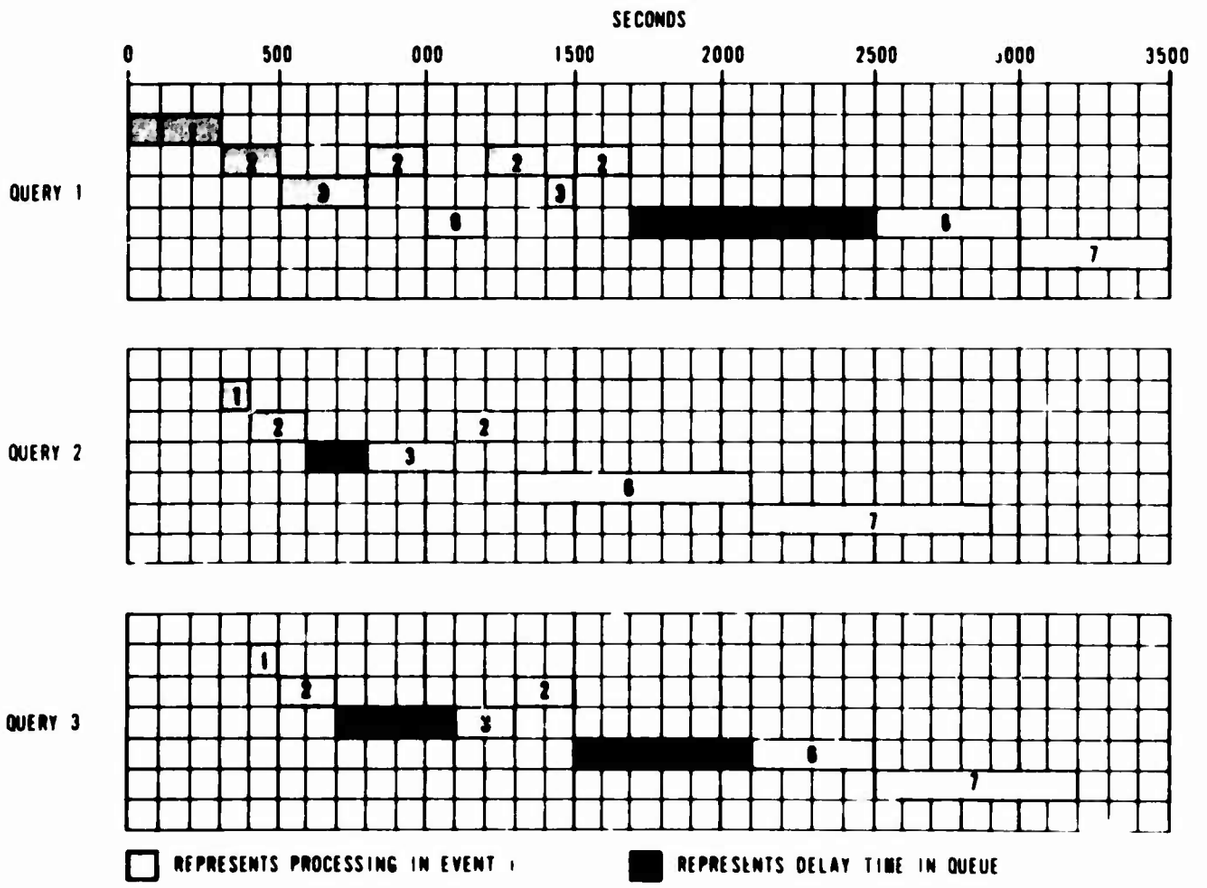


FIG. 18 INTEGRATED SEQUENCE OF EVENTS (CASE 2)

Reverse (Page-76-) Blank

APPENDIX A - COMPARSION TIME (T_{cn}):

A STUDY IN FORMULA DEVELOPMENT AND APPLICATION

Reverse (Page 78) Blank

APPENDIX A - COMPARISON TIME (T_{cn}); A STUDY IN FORMULA DEVELOPMENT AND APPLICATION

The utility of the different formulae for calculating processing times is dependent upon (1) how well the expressions reflect the resulting operations in the real world and (2) how well the engineer is able to estimate values for the different variables given in the equations. This section of the report gives some discussion of these two points, using the comparison time study as a presentation vehicle.

1. Formula Development

The time used in the logical comparison of a query statement with the items within the data base is a function of

(a) the speed of the central processor,

(b) the comparison program of the retrieval system,

(c) the logical expression and complexity of the query, and

(d) the volume and contents of the file examined in the search.

The following discussion explores how these aspects may interrelate in a general formula for calculating comparison time.

A gross expression of comparison time is

$$T_{cn} = \left(\frac{\text{time}}{\text{cycle}}\right)\left(\frac{\text{nr. cycles}}{\text{instruction}}\right)\left(\frac{\text{nr. instructions}}{\text{comparison}}\right)\left(\frac{\text{nr. comparisons}}{\text{record}}\right)\left(\frac{\text{nr. records ex.}}{\text{file search}}\right) \quad (1)$$

where:

- characteristics of computer
- parameters of IR program
- parameters of query logic
- file characteristic

In this expression, "time per cycle" represents the storage reference speed of the processor and can be considered to be a constant equipment characteristic for any given central processor. The remaining portions of the expression, however, may vary with different searches and, in fact, may vary from record to record. Different instructions may require different cycle times; moreover, the number and type of instructions executed in item comparison will normally vary from record to record. The comparison time required per file record is not likely to be constant; hence, the expression

$$\left(\frac{\text{nr. cycles}}{\text{instruction}}\right)\left(\frac{\text{nr. instruction}}{\text{comparison}}\right)\left(\frac{\text{nr. comparisons}}{\text{record}}\right)$$

must represent an average number of cycles per record to be meaningful in expression (1) above.

One method of determining an average number of cycles per record is to consider the path of the comparison operation through the logic of the retrieval program. For example, figure A1 illustrates one method of performing a simple retrieval comparison under a conjunctive or disjunctive query. If N represents the number of conjuncts (disjuncts) in a query and I the number of items tested in a file record, then the following table represents the number of cycles used under this retrieval logic.

	nr. cycles required to accept a record	nr. cycles required to reject a record
"AND" QUERY	$18I + 34$; $I = N$*	$18I + 26$; $1 \leq I \leq N$
"OR" QUERY	$18I + 32$; $1 \leq I \leq N$	$18I + 28$; $I = N$

* In this example, there are 40 cycles required in record manipulation and program initialization. The last item of the query requires 12 cycles in the comparison operation. Each of the previous I - 1 items requires 18 cycles; hence, $40 + 12 + 18(I - 1) = 18I + 34$ cycles per accepted record.

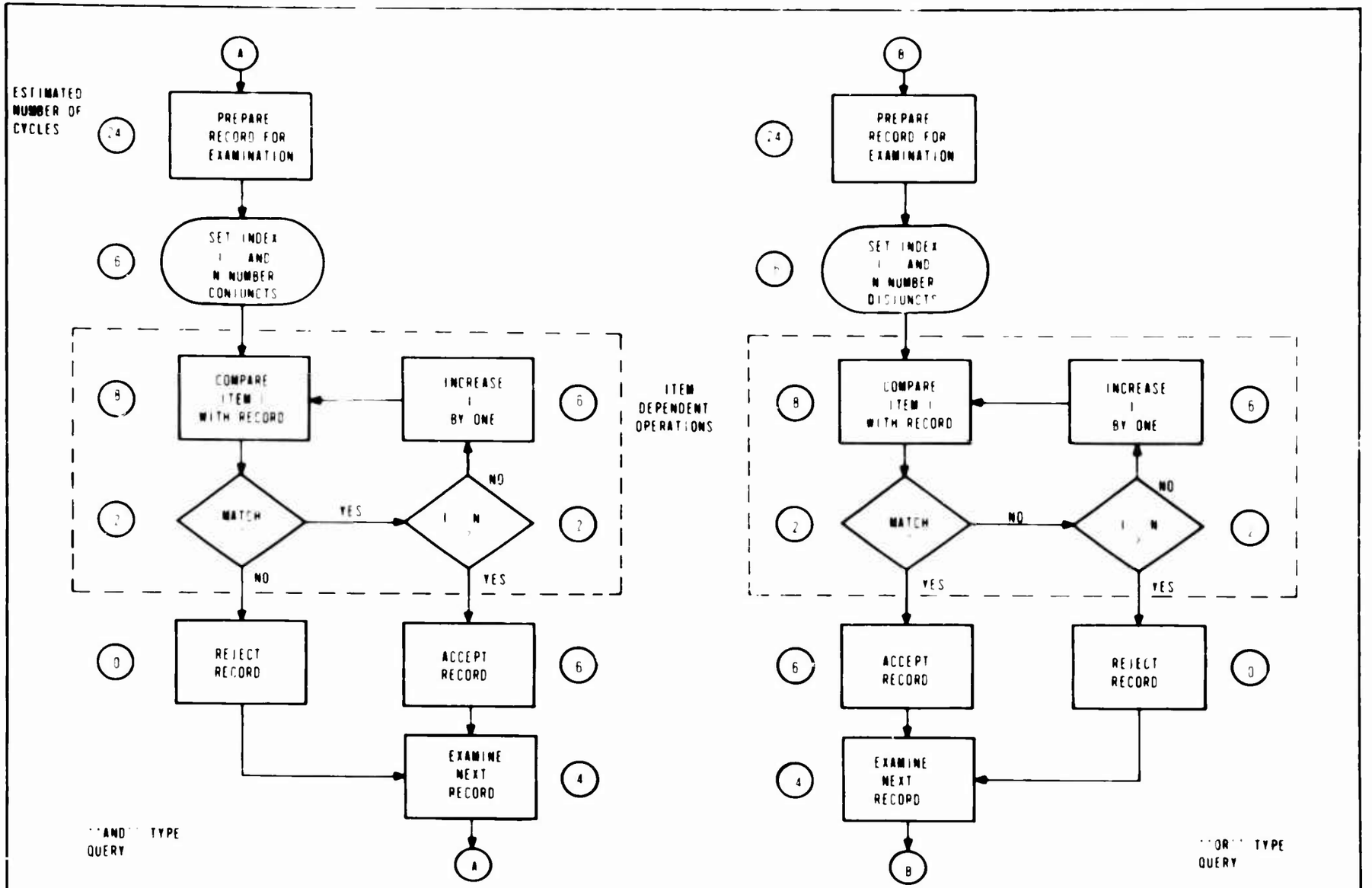


FIG A 1 EXAMPLE OF SIMPLE COMPARISON LOGIC FLOW CHART

In general, the average number of cycles required in the examination of a file record is a function of

(a) the logic of the query,

(b) whether the record is accepted or rejected in the comparison test, and

(c) the number of items tested.

Another (and somewhat more sensitive) expression of comparison time introduces these variables into the equation as

$$T_{cn} = (t \times 10^{-6}) \left[(C_i I + C_r) R + (C_i I' + C_r') R' \right] \qquad (2)$$

where

t is the processor speed in microseconds

C_i is the average number of cycles per tested item.

I (I') is the average number of items tested in each accepted (rejected) record.

C_r (C'_r) is a number of cycles associated with each accepted (rejected) record.

R (R') the number of examined records accepted (rejected) in a file search.

Each of the variables (C_i ... R') may take on different values, these values being associated with different queries or query classes considered in the simulation. It is contended that this expression of comparison time provides a reasonable approximation of the processing operation and that the variable values can be obtained from simple engineering studies of the information retrieval operation. It should be noted, however, that the calculated comparison time for a given file search does not necessarily add to the overall running time for a query. In some instances, all or part of the comparison operation may be imbedded

within the READ FILE operation. The additive portion of the comparison time may be expressed as non-negative values of

$$\left[T_{cn} - \left(\frac{\text{overlapped time}}{\text{block}} \right) \left(\frac{1}{\text{records/block}} \right) \quad R \right]$$

where:

$\overline{R}$ = R + R' and is the total number of records examined.

2. Formula Application

There are two general questions to be considered in applying time formulae to the simulation of system response time, i.e.,

(a) How can the variable values be determined?

(b) What are the effects of imprecise values on the time calculation?

The answers to both of these questions are, of course, fundamental to the basic problems of system engineering. For the moment, we shall beg the first question and only briefly explore the second. Both questions, however, will be treated in detail in subsequent reports.

Assume that an engineer has

(a) determined the equipment characteristics of the central processor; has

(b) estimated the software interaction during the comparison operation, and has

(c) determined the size of the data base,

the next step being to classify anticipated queries of the system (in terms of these parameters) for the calculations of T_{cn}. Examination of formula (2) for T_{cn} shows that there are three basic variables in the expression, i.e.,

the speed of the processor, the average number of cycles used per record and the number of records tested during the file search. Assume also that the engineer has adequately determined the cycle time of the processor and the number of file records examined in a search. The following discussion considers the problem of estimating the average number of cycles used per record.

Figure A2 illustrates a family of error curves relating the error in overestimating (or underestimating) the average number of cycles (E) with the cycle time (t) and the file volume ($\bar{R}$). In this illustration, the effect of the error is to produce less than 1 second deviation in the calculation of T_{cn} for the file search. For example, if the central processor has a cycle time of 25 microseconds, then for a file volume of 5,000 examined records the error in estimating the average number of cycles per record must be less than 8 in order to produce less than one second error in calculating T_{cn}. Conversely, if it is known that there is a possible error of $\pm$ E cycles per record, then, for a t-microsecond cycle time computer, the possible error in the calculation of T_{cn} can be denoted as ΔT, i.e.,

$$\Delta T = \pm \frac{E \cdot t \quad \bar{R}}{10^6} \text{ seconds}$$

For example, if $E = \pm 20$ cycles per record

$t = 25$

$\bar{R} = 10,000$ records

the possible error in calculating T_{cn} is

$$\Delta T = \frac{\pm 20 \cdot 25 \cdot 10^4}{10^6} = \pm 10 \text{ seconds.}$$

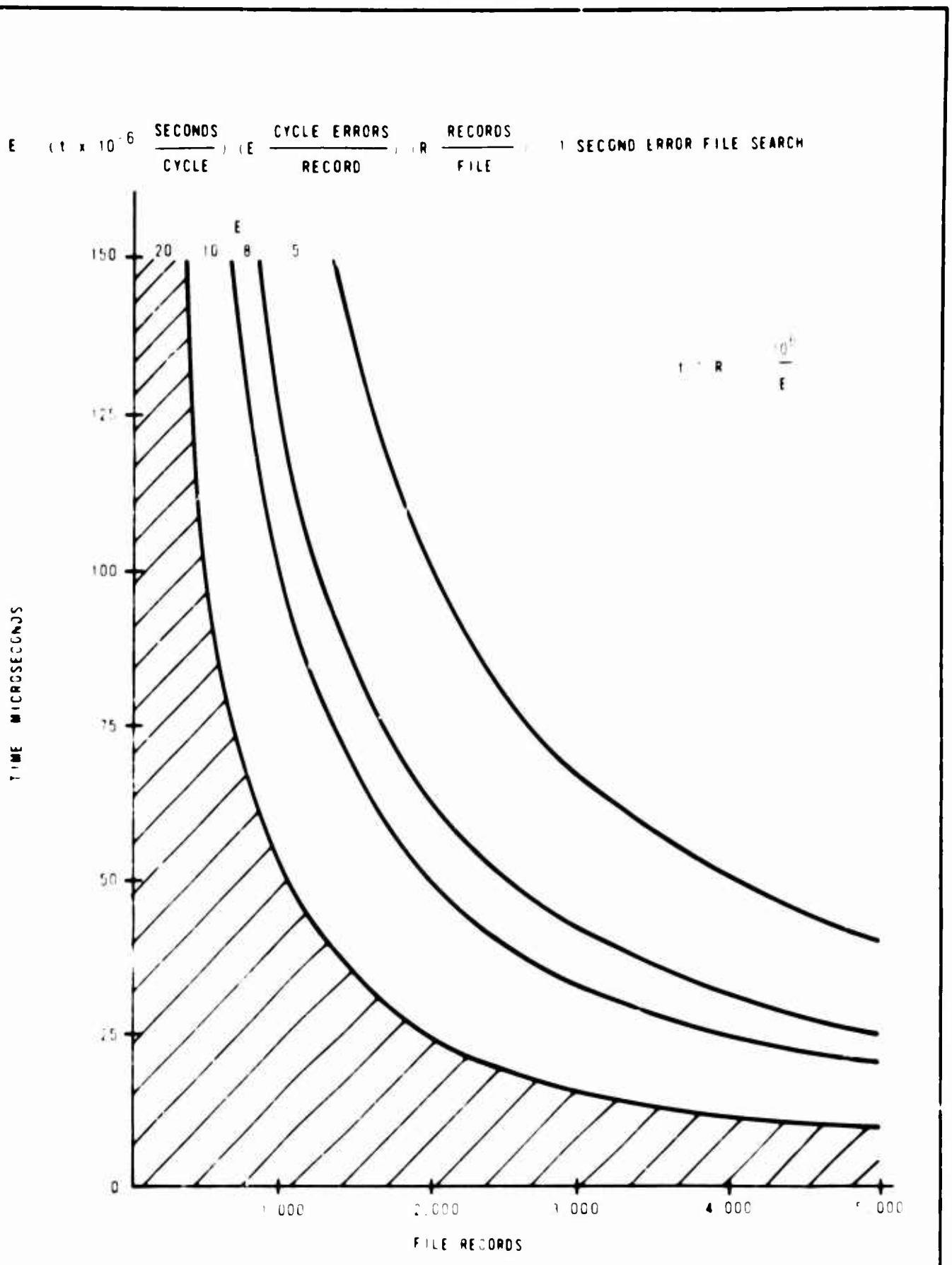


FIG. A 2 RELATIONSHIP BETWEEN SPEED OF CENTRAL PROCESSOR, FILE VOLUME AND ERROR IN CALCULATING T_{CN}

Determination of E is somewhat complex, but can be done as follows. Let

$$T_{cn} = (T_{cn})_A + (T_{cn})_R$$

where

$$(T_{cn})_A = (t \times 10^{-6}) (C_i I + C_r) R,$$

i.e., the comparison time required to examine and accept the relevant file records; and

$$(T_{cn})R = (t \times 10^{-6}) (C_i I' + C_r') R',$$

i.e., the comparison time required to examine and reject the non-relevant file records. The effect of errors in estimating the values of the variables can be examined by considering an over-estimation of C_i, I and C_r in calculating $(T_{cn})_A$.

Suppose the values for t and R are accurate, but that C_i, I and C_r have been overestimated and are erroneously given as C_i^*, I* and C_r^* where

$$\left.\begin{aligned} C_i^* - C_i &= \Delta C_i \\ I^* - I &= \Delta I \\ C_r^* - C_r &= \Delta C_r \end{aligned}\right\}$$

The cumulative error in calculating $(T_{cn})_A$ is denoted as ΔT_A, i.e.,

$$(T_{cn})_A + \Delta T_A = (t \times 10^{-6}) (C_i^* I^* + C_r^*) R \qquad (3)$$

and (T_{cn}) can be expressed as

$$(T_{cn})_A = (t \times 10^{-6}) \left[(C_i^* - \Delta C_i)(I^* - \Delta I) + (C_r^* - \Delta C_r) \right] R$$

$$= (t \times 10^{-6}) \left(C_i^* I^* - C_i^* \Delta I - \Delta C_i I^* + \Delta C_i \Delta I + C_r^* - \Delta C_r \right) R$$

$$= (t \times 10^{-6}) \left(C_i^* I^* - C_i^* \Delta I - \Delta C_i I^* + \Delta C_i \Delta I + C_r^* - \Delta C_r \right) R \quad (4)$$

Subtracting (4) from (3) and collecting terms yields

$$\Delta T_A = (t \times 10^{-6}) \left[(C_i^* - \Delta C_i) \Delta I + \Delta C_i I^* + \Delta C_r \right] R \quad (5)$$

where C_i^* and I^* are estimated values and ΔC_i, ΔI and ΔC_r are maximal deviations from exact variable values.

Similarly, an expression for the cumulative error in calculating $(T_{cn})R$ is

$$\Delta T_R = (t \times 10^{-6}) \left[(C_i^* - \Delta C_i) \Delta I' + \Delta C_i (I')^* + \Delta C_r' \right] R'. \quad (6)$$

Now, the possible error in calculating T_{cn} is ΔT where

$$\Delta T = \pm \frac{E \cdot t \cdot \bar{R}}{10^6} \text{ seconds.}$$

Moreover,

$$\text{MAX } \Delta T = \Delta T_A + \Delta T_R$$

hence,

$$\text{MAX } E = \frac{10^6}{t \cdot \bar{R}} \left[\Delta T_R + \Delta T_A \right].$$

The expansion of this expression for E is straightforward from (5) and (6); however, it is possible to obtain useful results without considering the entire expression of E. The following examples illustrate some applications of the expressions for ΔT_A and ΔT_R.

EXAMPLES·

An IR system is exemplified by the logic flow chart given in Figure A1. An engineering study of the system produces the following data:

t = 12 microsecond cycle time

C_i = 18 cycles ($\Delta C_i = 0$)

C_r = 32 or 34 cycles depending upon the query

$C_{r'}$ = 26 or 28 cycles depending upon the query.

Additionally, it is known that the file contains 22,440 records and that the maximum output for any query is 200 records. A simple, though not precise, expression of comparison time for this system is

$$T_{cn} = (12 \times 10^{-6}) [(18 \cdot I + 33)R + (18 \cdot I' + 27)R']$$

where $C_r^* = 33$ and $(C_r')^* = 27$ are substituted as estimated values for C_r and C_r'. It is further estimated that when the number of items tested in a record is a variable (e.g., $1 \leq I \leq N$; where N is the number of items in the query), the average number tested per record will be $\frac{N}{2}$; elsewhere I (or I') = N.

Under AND and OR queries, the maximum error deviation in the estimated values are

AND	OR
$\Delta C_i = 0$	$\Delta C_i = 0$
$\Delta C_r = 1$	$\Delta C_r = 1$
$\Delta I = 0$	$\Delta I = \pm \frac{N}{2}$
$\Delta C_r' = 1$	$\Delta C_r' = -1$
$\Delta I' = \pm \frac{N}{2}$	$\Delta I' = 0$

The maximum error in calculating $(T_{cn})_A$ for an AND query can be found from (5), i.e.,

$$\Delta T_A = (12 \times 10^{-6}) [(18)\ 0 + 0 \cdot N - 1]\ 200 = -.0024 \text{ seconds}$$

and from (6) the maximum error in calculating $(T_{cn})_R$ for an AND query is

$$\Delta T_R = (12 \times 10^{-6}) [(18) (\pm \frac{N}{2}) + 0 \cdot \frac{N}{2} + 1]\ 22,240 = (1 \pm 9N)\ (.27) \text{ seconds.}$$

A comparison of ΔT_R with ΔT_A shows that ΔT_R may be significant for some values of N while ΔT_A is not. The significance of ΔT_R can be examined by calculating $(T_{cn})_R$, i.e.,

$$(T_{cn})_R = (12 \times 10^{-6}) (18 \cdot \frac{N}{2} + 27)\ 22,240 = (27 + 9N)\ (.27) \text{ seconds.}$$

Figure A3 gives the graph of $(T_{cn})_R$ and $(T_{cn})_R + \Delta T_R$. The upper

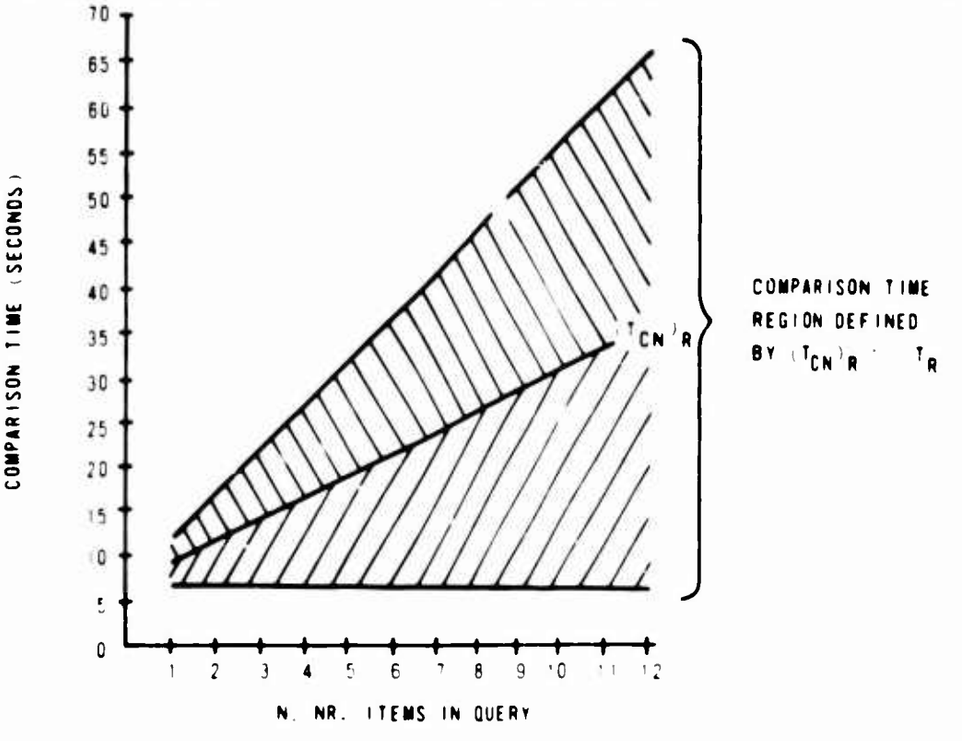


FIG. A-3 GRAPH OF $(T_{CN})_R$ AND $(T_{CN})_R$ T_R FOR EXAMPLE SYSTEM

and lower boundaries represent maximum possible error; that is, in the rejection of 22,240 records, the estimate of $\frac{N}{2}$ tests per record was wrong by a factor of $\frac{N}{2}$. Normally, such would not be the case, and

the error would range between 0 and $\frac{N}{2}$. Careful analysis of the data base and query classes should yield close approximations of the number of items tested under each query class for each query type.

In this example system, the OR type request has the same kind of comparison time graph except the significant aspect is ΔT_A instead of ΔT_R.

APPENDIX B - PROGRAM LISTING OF THE SEQUENCE INTEGRATOR SUBROUTINE

Reverse (Page 92) Blank

APPENDIX B - PROGRAM LISTING OF THE SEQUENCE INTEGRATOR SUBROUTINE

The SEQUENCE INTEGRATOR is an algorithm which will time-wise integrate the processing steps of a set of queries. Each processing step is defined as being composed of three elements: the step number, an event number, and an event service time. The algorithm will determine the total processing time of the i^{th} query (T_i) by considering equipment availability, the processing steps of all queries, and so forth. As a result, the total processing time for the set of queries can be expressed as equal to

$$\max_i \left\{ T_i \right\}.$$

For example, if two queries (A and B) are started at the same time and query A finishes at time 47 and query B finishes at time 34, then the total processing time is given by $\max \left\{ 47, 34 \right\}$ which means that both queries were processed in 47 time units.

Other data, such as the sum of the delay times for each query, is also available for use. The latter information can be found as follows:

Let

D_i = total delay time for query i

t_{ij} = service time, j^{th} step, i^{th} query

T_i^* = minimum processing time required for query i

n_i = number of processing steps for query i.

Then, we have

$$T_i^* = \sum_{j=1}^{n_i} t_{ij}$$

and, hence,

$$D_i = T_i - T_i^*.$$

The remaining portion of this section contains the logic flow chart, and program listing of the Sequence Integrator Subroutine, along with the symbol table.

Table B1 - Sequence Integrator Subroutine Symbol Table

SYMBOL	MEANING
A	Dummy variable (used to indicate earliest service unit availability time)
AT(I, J)	Availability time of the J^{th} unit, I^{th} event
C	Dummy variable (used to indicate earliest executable query time)
ET(I)	Next event time for query I
I	Dummy variable
J	Dummy variable
K	Dummy variable
L	Dummy variable
M	Dummy variable
MM	Dummy variable
N	Dummy variable
NE	Number of events
NQ	Number of queries
NRS(I)	Number of remaining steps for query I
NS(I)	Total number of steps for query I
NSU(I)	Number of service units for I^{th} event
QET(I, J, 1)	Event number, I^{th} query, J^{th} step
QET(I, J, 2)	Service time, I^{th} query, J^{th} step
QET(I, J, 3)	Service indicator, I^{th} query, J^{th} step (0 means J^{th} step not serviced and 1 means J^{th} step serviced)
SUMQET	Dummy variable (used to determine total processing time required for a query)
SUT (I, J)	Usage time of J^{th} unit, I^{th} event

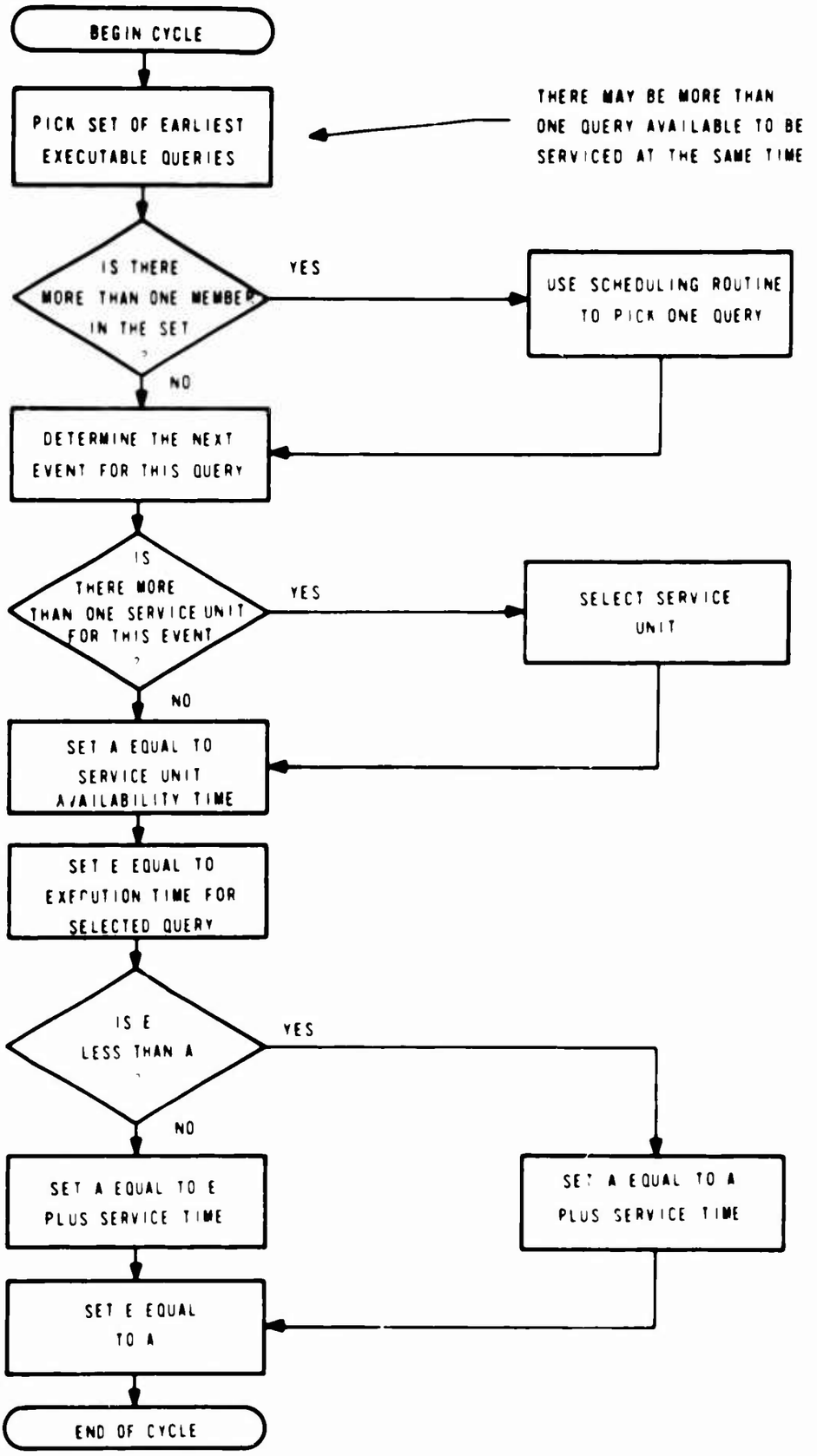


FIG. B-1 SEQUENCE INTEGRATOR SUBROUTINE LOGIC

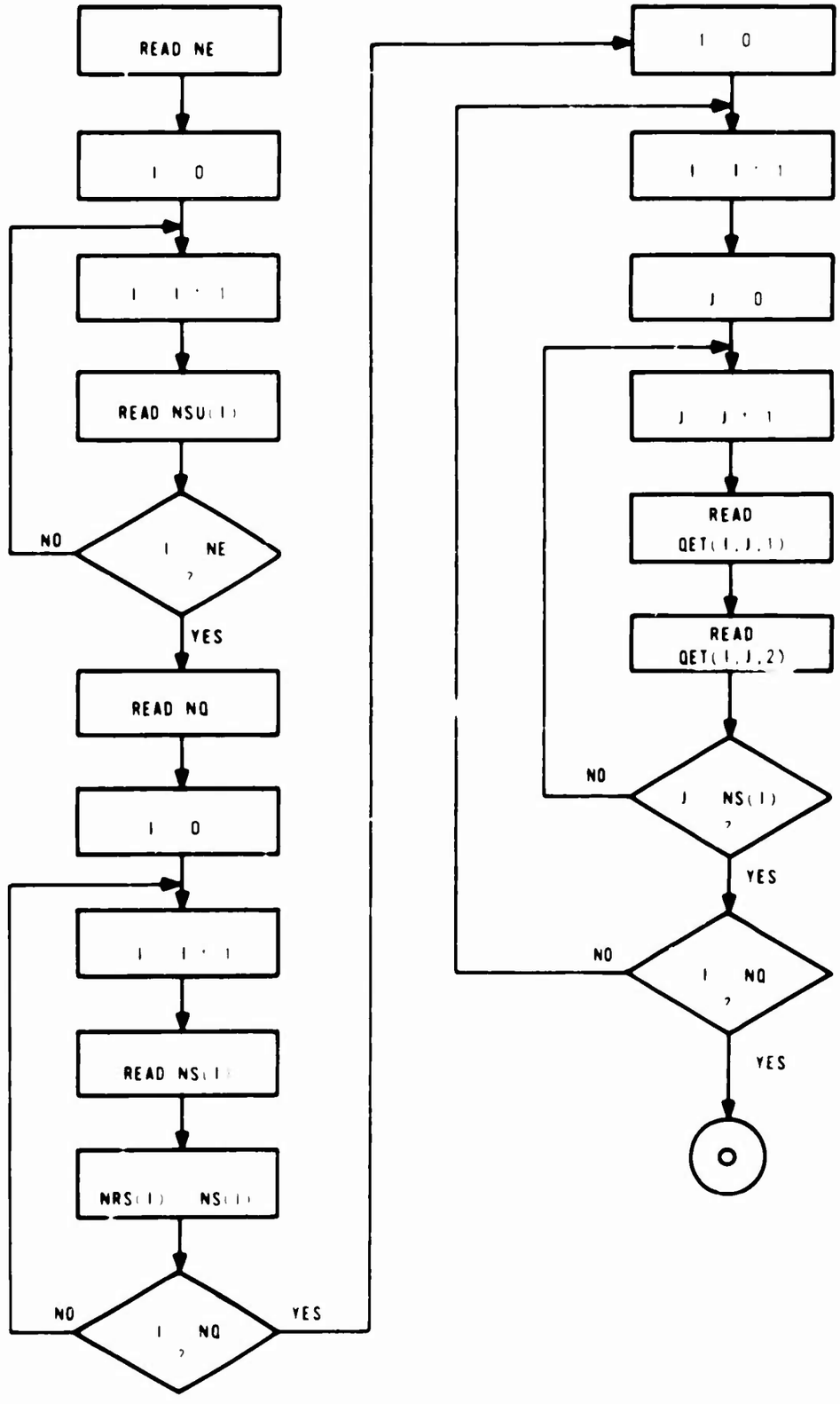


FIG. B-2 SEQUENCE INTEGRATOR SUBROUTINE FLOW CHART

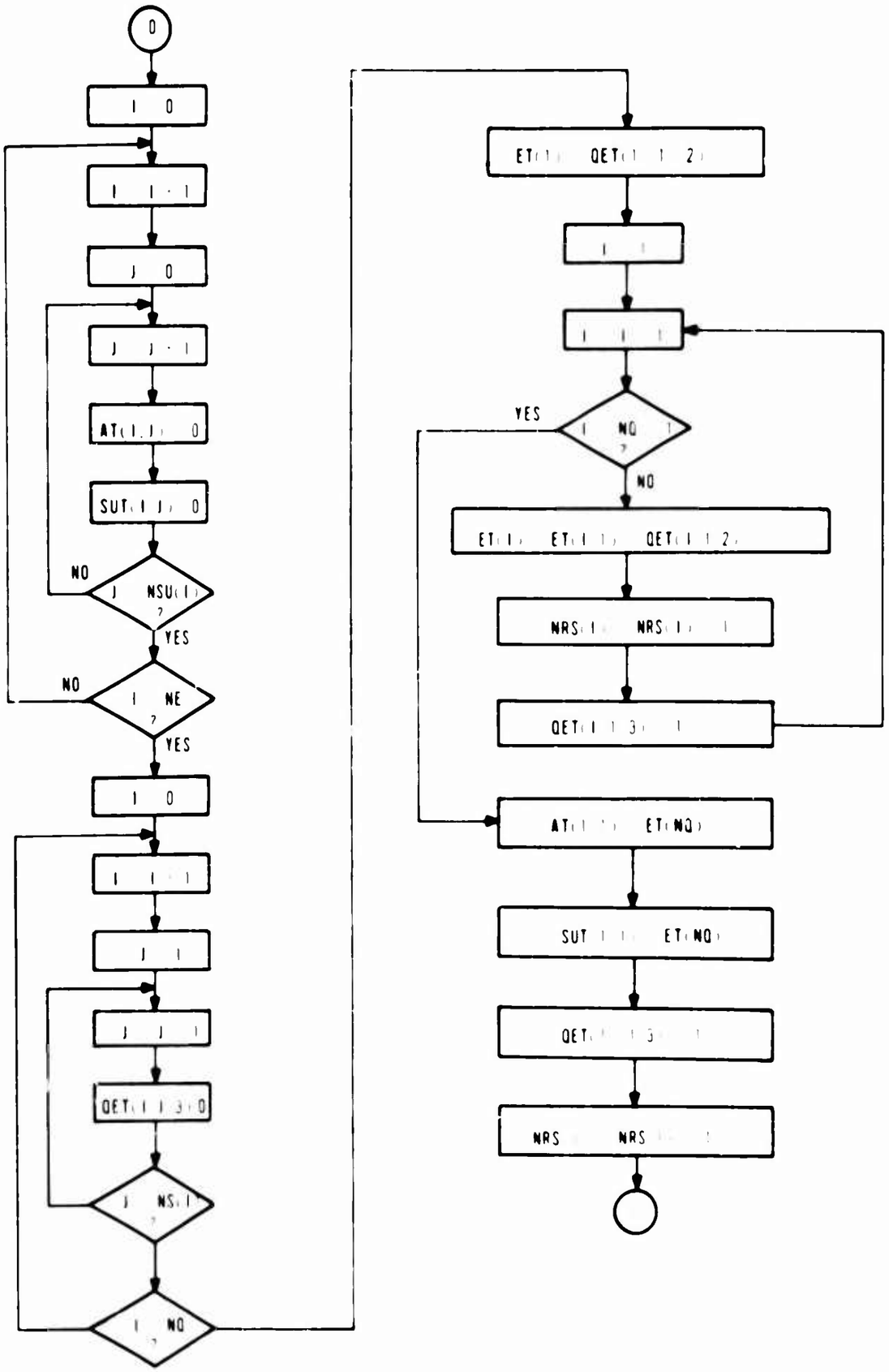


FIG B-2 SEQUENCE INTEGRATOR SUBROUTINE FLOW CHART (CONT D)

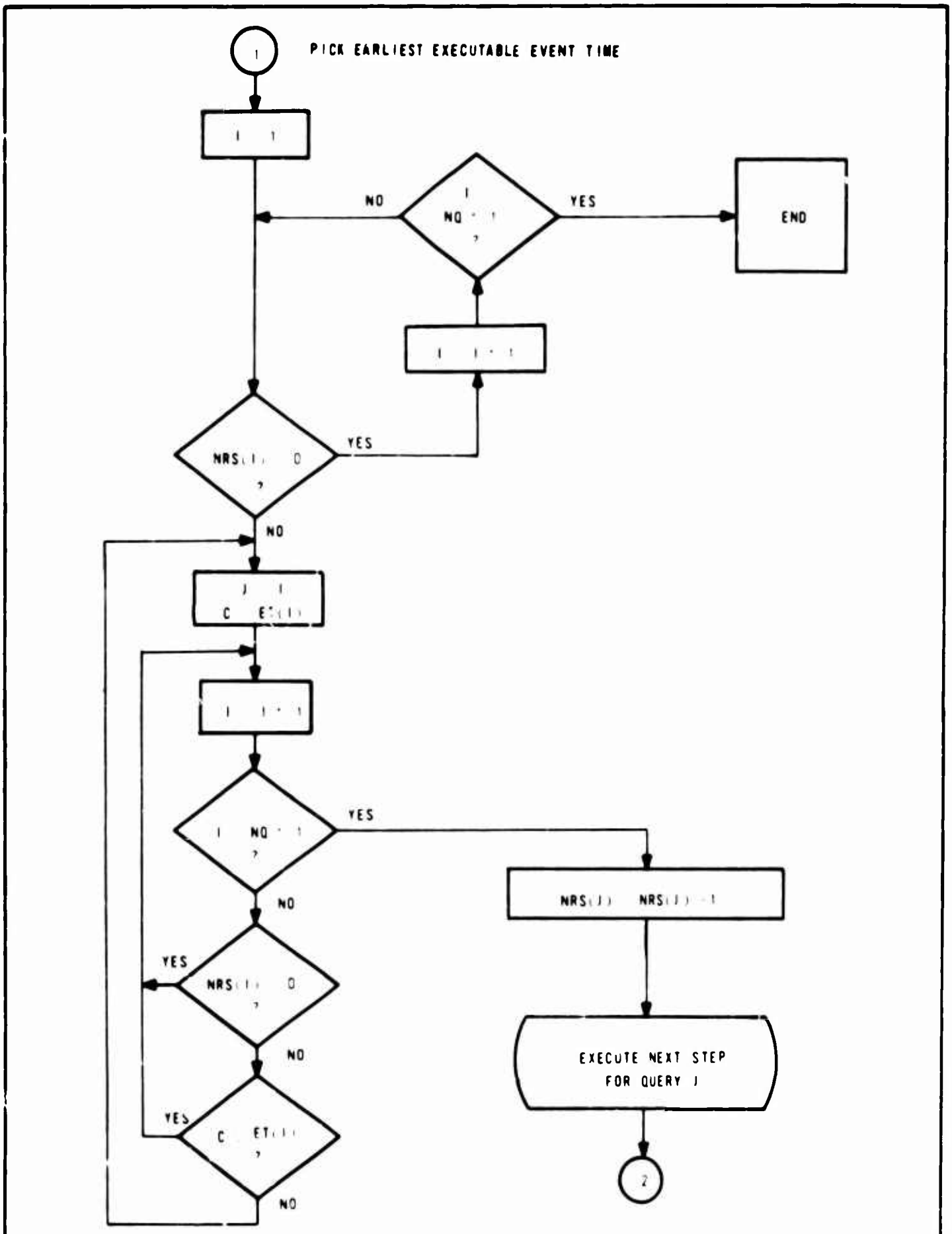


FIG. B-2 SEQUENCE INTEGRATOR SUBROUTINE FLOW CHART (CONT'D)

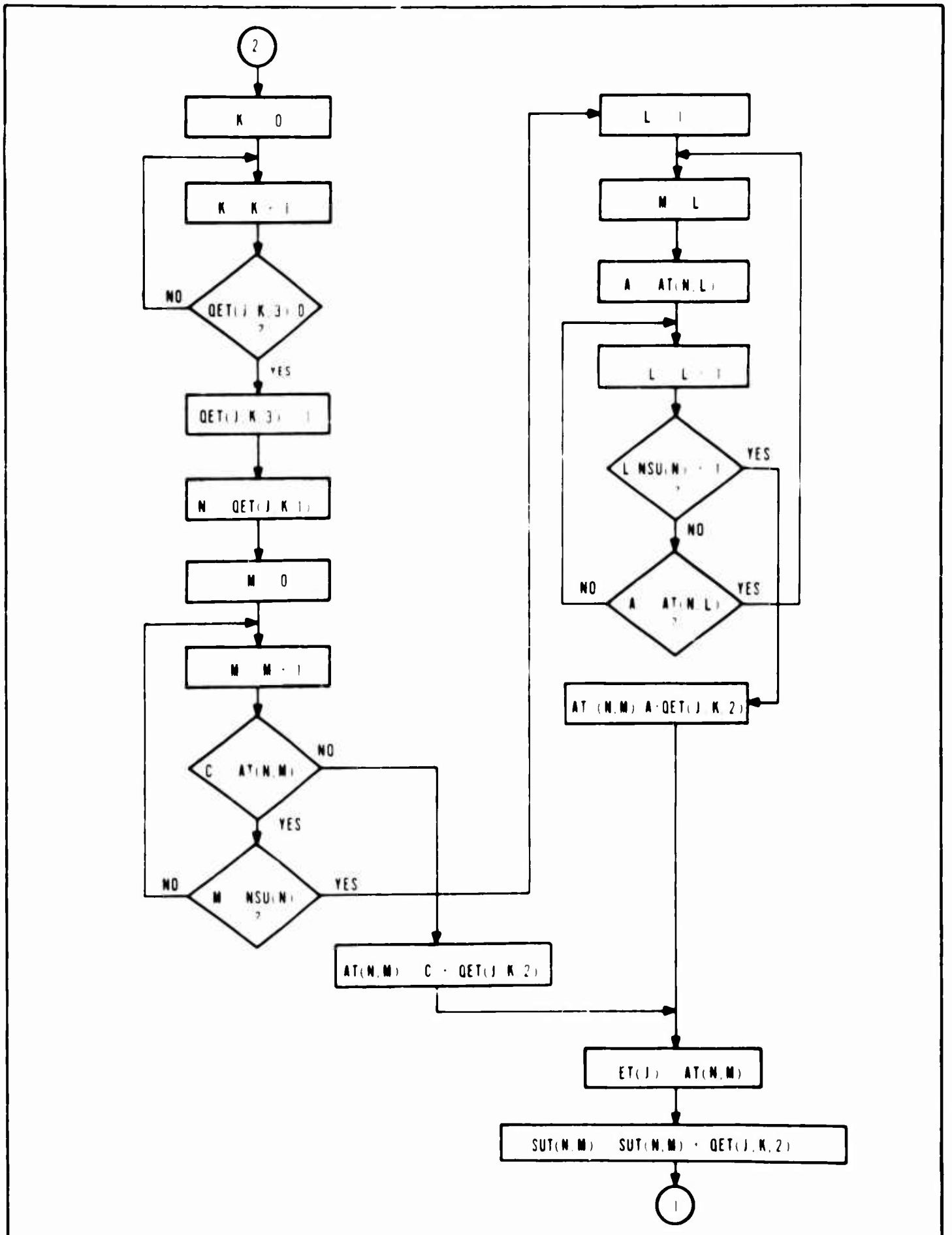


FIG B-2 SEQUENCE INTEGRATOR SUBROUTINE FLOW CHART (CONT'D)

```
C       SEQUENCE INTEGRATOR
        DIMENSION AT(10,5),ET(10),NRS(10),NS(10),NSU(10),QET(10,20,3),
       1SUT(10,5)
  100 FORMAT(I2)
  101 FORMAT(10I2)
  102 FORMAT(I2)
  103 FORMAT(I2)
  104 FORMAT(I2,2X,F8.2)
  200 FORMAT(1H ,I2,3X,F8.2)
  201 FORMAT (1H1,5HQUERY,5X,20HREQD PROCESSING TIME,5X,15HCOMPLETION TI
      1ME)
  202 FORMAT (1H,I3,12X,F8.2,15X,F8.2)
  203 FORMAT(1H0,27X,12HSERVICE UNIT)
  204 FORMAT(1H ,20X,1H1,10X,1H2,10X,1H3,10X,1H4,10X,1H5)
  205 FORMAT(1H ,5HEVENT)
  206 FORMAT(1H ,2X,I1,9X,5F11.2)
    1 READ 100,NE
      READ 101,(NSU(I),I=1,NE)
      READ 102,NQ
      DO 2 I=1,NQ
      READ 103,NS(I)
    2 NRS(I)=NS(I)
      DO 3 I=1,NQ
      K=NS(I)
      DO 3 J=1,K
    3 READ 104,QET(I,J,1),QET(I,J,2)
      DO 4 I=1,NE
      K=NSU(I)
      DO 4 J=1,K
      SUT(I,J)=0
    4 AT(I,J)=0
      DO 5 I=1,NQ
      K=NS(I)
      DO 5 J=1,K
    5 QET(I,J,3)=0
      ET(1)=QET(1,1,2)
      DO 6 I=2,NQ
      ET(I)=ET(I-1)+QET(I,1,2)
      NRS(I)=NRS(I)-1
    6 QET(I,1,3)=1
      AT(1,1)=ET(NQ)
      SUT(1,1)=ET(NQ)
      QET(1,1,3)=1
      NRS(1)=NRS(1)-1
C                                     BEGIN CYCLE
    7 I=1
    8 IF(NRS(I))10,9,10
    9 I=I+1
      IF(I-NQ-1)8,25,8
   10 J=1
      C=ET(1)
   11 I=I+1
      IF(I-NQ-1)12,14,14
   12 IF(NRS(I))13,11,13
   13 IF(C-ET(I))11,11,10
   14 NRS(J)=NRS(J)-1
```

```
      K=0
   15 K=K+1
      IF(QET(J,K,3))15,16,15
   16 QET(J,K,3)=1
      N=QET(J,K,1)
      M=0
   17 M=M+1
      IF(C-AT(N,M)) 18,29,29
   18 IF(M-NSU(N))17,19,17
   19 L=1
   20 M=L
      A=AT(N,L)
   21 L=L+1
      IF(L-NSU(N)-1)22,23,22
   22 IF(A-AT(N,L))21,21,20
   23 AT(N,M)=A+QET(J,K,2)
   24 ET(J)=AT(N,M)
      SUT(N,M)=SUT(N,M) + QET(J,K,2)
      PRINT 200,J,AT(N,M)
      GO TO 7
C                                          END CYCLE
   25 PRINT 201
      DO 27 I=1,NQ
      SUMQET=0
      MM=NS(I)
      DO 26 J=1,MM
   26 SUMQET=QET(I,J,2)+SUMQET
   27 PRINT 202,I,SUMQET,ET(I)
      PRINT 203
      PRINT 204
      PRINT 205
      DO 28 I=1,NE
      K=NSU(I)
   28 PRINT 206,I,(SUT(I,J),J=1,K)
      PAUSE
      GO TO 1
   29 AT(N,M)=C+QET(J,K,2)
      GO TO 24
      END
```

APPENDIX C

SELECTED ABSTRACTS ON THE SIMULATION AND MODELING OF INFORMATION RETRIEVAL SYSTEMS

Reverse (Page 104) Blank

APPENDIX C

SELECTED ABSTRACTS ON THE SIMULATION AND MODELING OF INFORMATION RETRIEVAL SYSTEMS

A limited examination of current DDC reports relating to studies, simulation, and modeling of information systems has resulted in the following summaries. While the reports show that many unique and valuable techniques have materialized, the area of the user remains a prime problem, having received perhaps the least emphasis. The semantic and linguistic aspects of information retrieval systems also lend themselves poorly to the rigidity of models and model techniques for which claims often lack empirical support. However, the value of simulation and modeling as research tools and techniques for information retrieval system studies has been demonstrated and the accomplishments, some of which are summarized here, are well worth the attention of the researcher, designer or evaluator.

Reverse (Page 106) Blank

Project SHARP

(A Case Study)

Originator	Date	Customer	Authors	Source of Information
Herner & Co.	(No Date)	U.S. BUSHIPS & Air Force Office of Scientific Research	Saul Herner, F.W. Lancaster, Walter F. Johanningsmeier	AD 608 743, A Case Study in the Application of Cranfield System Evaluation Techniques

SUMMARY

Through the use of Cleverdon's methods in the ASLIB Cranfield Project, an effort was made to evaluate and to maximize the effectiveness of a computerized information retrieval system based on the Engineers Joint Council (EJC) system of role indicators and links. A typical document was indexed under 10 to 15 terms which with role combinations became somewhat more than 20. Under the project, 750 documents (classified and unclassified) were indexed by two indexers who had taken the one-week course at the Battelle Memorial Institute in the EJC system. The authority for subject indexing was BUSHIPS' Thesaurus of Descriptive Terms and Code Book, similar to the Thesaurus of ASTIA Descriptors. The indexing process was selective, that is, only central and important concepts in a report were indexed. Descriptors were translated into an alpha-numeric code for the computer (IBM 7090) and a series of test searches were made.

Retrieval effectiveness was expressed in terms of relevance and recall ratios. Relevance ratio for a search product is defined as the proportion of relevant selected documents to the total retrieved documents. Recall ratio is defined as the proportion of selected relevant documents to the total set of known relevant documents in a collection.

The search failures and less than optimal results were analyzed in terms of indexing faults, searching faults and system faults. The test was based on 50 searches, with the test of 150 questions coming from the scientists and engineers of The Bureau of Ships. The documents produced from each search were submitted to the compilers of the questions, and each compiler was asked to decide whether the results were responsive or not, thus furnishing a basis for the relevance ratios. The recall ratios were determined by actually doing a total

check of the specimen collection to locate all documents having any possible relevance to 10 of the test questions. Seemingly relevant documents not uncovered in the original searches were submitted to the question compilers for relevance assessment. Typical relevance and recall ratios at Cranfield were around 20 percent relevance and 80 percent recall, but in this case, the figures were:

	Relevance	Recall
Source + relevance A documents (as relevant as source documents)	29.7%	68.3%
Source + other relevant documents	54.3%	53.8%

The reasons for nonrelevant retrieval of documents fall under the categories of searching errors, indexing errors and system failures. Search errors gave the greatest "noise," with the failure to select crucial terms being a major problem.

On the matter of failure to retrieve documents relevant to search questions, the results were different in that indexing errors contributed more heavily to the cause of failure, also, system inadequacies were significant factors. Errors in searching procedures were still important in recall failures, but not so much as in spurious documents in the search product. Additional tabulations are contained on the searches and further reasons given for the ratios of recall and relevance. Some basic causes for failure are:

1. Searcher did not make selection specific enough through the use of search terms.
2. Searcher failed to exhaust all possible subjects and subject combinations.
3. Indexers omitted important concepts.
4. Not all appropriate roles were applied to descriptors under which the documents should have been retrieved.
5. Thesaurus used did not establish relationships among vital terms, thus causing them to be missed on both searching and indexing sides.
6. On the retrieval of nonrelevant documents, the searcher failed to use an important qualifying subject in the search program.

The conclusions of the authors are that relevance and recall ratios cannot be construed as figures of merit, that is, they do not tell really whether the system is good or bad in any absolute sense. There is some required trade-off between purity and completeness because one is not easily optimized without some expense to the other. The greater use of links will diminish the noise level, and some increase in recall and decrease in relevance can be achieved through fewer role indicators. But the most important conclusion, in the words of the authors, is "No evaluation technique can tell us what we want or need. This we have to decide for ourselves."

COMMENTS

The evaluation procedures developed by the Cranfield Project on aeronautical data have been subsequently applied to other systems and other data, of which this report is an example. While the statistics on the criteria of recall and relevance are valuable as with respect to trade-off, the Cranfield methods do require strict control and total checks on the system. The larger the system being evaluated the greater the control required for a successful and complete evaluation. Also, the larger the system the more expensive the evaluation becomes, especially on a live-test basis.

The subjective factors on which the Cranfield techniques rely, particularly the matter of relevance, have been a point of contention or rejection on one hand and of an acceptably good approach on the other. Whereas evaluation techniques did not exist before, they do now although subjectively based. Credit must be given to this formalization of difficult subjective factors into statistics, a prime step toward total evaluation.

Reverse (Page 110) Blank

Simulation of Time-Sharing System

Originator	Date	Customer	Author	Source of Information
SDC	Dec. 1964	Prof. paper presented at Inst. of Management Sci. Meeting, San Fran. 3-5 Feb. 1965	Gerald H. Fine and Paul V. McIsaac	AD 611 868, Simulation of a Time-Sharing System.

Summary of Method

The use of simulation techniques at SDC in the analysis of time-share system operations is described. Time-Sharing-System simulation was begun in early 1964 about a year after the SDC Time-Sharing System first became operational. There are two purposes of the simulation

1. To provide a vehicle for the prediction and evaluation of the effects of proposed system design and computer configuration modifications.
2. To serve as a tool for studying the potentiality and limitations of the time-share concept in terms of applicability to various work-load environments.

Illustrations of the SDC Time Sharing-System (TSS) are given. The equipment configuration of the SDC Command Research Laboratory consists of a Q-32 central processor, a large fast general-purpose digital computer with many units of peripheral equipment, including a PDP-1 computer for controlling real-time inputs and outputs. Except for displays, these devices may be operated from within SDC as well as from geographically remote stations. As of December, 1964, the system included 27 local teletypes and 6 display consoles located within SDC, plus a capacity to handle simultaneously up to 6 remote teletypes. There were no remote computers linked to the system at that time.

The programs may be "object" (user-written) programs or service programs provided by the system. Users who want to take advantage of the system communicate with the TSS Executive from a teletype station, with each user's

requirements being serviced in a round-robin fashion. Each user to be serviced is allocated a specific quantum of time, controllable by a programmable "quantum clock" which interrupts the computer after a set interval. When a user comes due for a turn of service, the object program currently in core memory is restored to its assigned drum area, and the program to be operated is read into core memory from its assigned drum area, with the transfer being referred to as a "swap." The swap is omitted if a program due for a turn happens to be in core memory from a previous turn of service.

Some remote users are Carnegie Institute of Technology, Los Angeles Police Department, Massachusetts Institute of Technology, The Rand Corporation, Stanford Research Institute, Stanford University, University of California, and the Veterans Administration.

For the simulation of the above system, it was necessary to set up the input parameters. The work-load environment was created by a job generator using Monto Carlo techniques to produce a series of entities called jobs with an exponentially distributed interarrival time. Each job consisted of a set of values describing its size, service requirements and the like, with up to five job classes generated in any desired mix. The mean arrival rate, mix proportions and job-class descriptions were specified. There are eight variables to each job description, and they may be specified either as fixed values or as the means of normal distributions. The generator is made operable with "work day begin and end" parameters. Jobs queue up as backlog and are assigned eventually to system channels, where they are logged in, loaded and pass through repeated service cycles. These operations in the simulation are handled after the real system with respect to quantum allotment, idle cycle duration, and overhead factors. The environment configuration is also treated with respect to number of channels, core and drum size, access and word transfer rates and so on. Job arrivals, computer malfunctions, service requests, etc., all constitute events occurring in simulated time. The simulator works on the next event basis.

The simulator outputs to teletype and to tape. The running accounts to the teletype of the major events permits monitoring of the run by the user, while the more detailed data goes to tape. System operation is measured by producing demand and response-cycle distributions, where demand is defined as the "number of operable users in any one round robin," and response is the time taken to process the round robin. "Worst" response cycles are also recorded. Computer

use is measured in terms of percentages of time spent on swapping, service, overhead and idling.

Comments

This study is probably one of the few simulations that have been made on a time-sharing system, in this case performed by the same firm that built the system. There may be some distinct advantage in simulating one's own system but SDC admits that the use of the simulator did not come easy. The first simulation program was written in a procedure oriented language with standard library routines for output, and it ran so slowly (20 to 45 minutes for a day's simulation) that is had to be abandoned. The next simulator was written in a machine language, requiring 2 6 minutes for a day's simulation.

The simulator findings were interesting. The mean demand (number of users requiring service at scheduling time) is high when users cannot complete service in a turn or two but improves quickly as they receive larger allotments of processor time. But after a point, the additional improvement is not sufficient to warrant the possibly poor response due to an occasional long-service requirement. The computer use on percentage of time devoted to processing is similarly reflected, since less swapping and overhead are required when users are serviced to completion. SDC concludes that the simulator seemed to show the same noisy characteristics of the real system, in other words, produced data (given in the document as typical demand versus quantum curve, etc.) comparable to subjective observations upon the system. Graphs and other appropriate illustrations are given for this information.

An alternative scheduling technique was attempted. The older algorithm treated all job types identically, but with a new algorithm a distinction was made between jobs requiring small but infrequent amounts of service with fast response, and those needing considerable processing time with relatively little user interaction. All aspects of system performance improved: response, throughput, turnaround time and utilization. The new scheduling technique was found superior, on the basis of visual analysis of recordings and user feedback. SDC, however, mentions that the size of the system sample was limited, and conclusions based on the findings must remain indicative.

SDC also tried a Direct Search Method to deal with the great number of input parameters and wide range of possible values with each parameter. This was really a separate effort from the simulator operation itself, and the equations for optimization of search are given. Findings to date are said to be promising.

Applications to HOIST IR Model

While the HOIST IR model is not at the stage of optimization or refinement, since it is a general model of the retrieval process, it is likely that optimization will soon be of concern since a Time-Sharing retrieval system requires greater optimization of search than is required with nonsharing systems. The number of users for the Time-Sharing system seems to indicate that this type of retrieval system may be the best and most efficient use of the retrieval process, especially with the advantage of remote operation. The HOIST IR model is presently time-oriented, and time remains the number one concern in the SDC Time Sharing system.

Methodology for Computer Simulation

Originator	Date	Customer	Author	Source of Information
Mass. Inst. of Technology	Oct. 1964	Advance Res. Projects Agency, Dept. of Defense, under Office of Naval Research Contract	Martin Greenberger	AD 609 288, A New Methodology for Computer Simulation

Summary of Method

The goals of Project MAC at M.I.T. are to make the computer more generally useful to the researcher. Project MAC is the simulation of time-shared multi-user computer operation, a way of using the computer to produce a reasonable likeness of the behavior of the system under study. The likeness is a scaled-down abstraction of the real system in the form of a dynamic model. The model is based on the simulator's concept of what the key elements of the system are and of how they operate and interact.

The MIT OPS-2 is a research tool linking the user with the computer in a laboratory environment to make mutual interaction simple and powerful. OPS-2 is an on line system, being at the researcher's side, in effect, like his manuals, journals, notebook and telephone. There is no sharp dividing line between the phases of data analysis, formulation, programming, running and validation in simulation. The OPS system is relatively free of rules and formats. The user creates his own language and his own conventions. The emphasis is upon his latitude to express the problem in its natural terms and to be inventive.

The basic structure of the OPS system is a body of data located in common storage, and there is a set of operators which operate this data. The data consists of lists, multidimensional arrays, and single elements. Reference to the data is symbolic, and an index of symbols and dimensions is incorporated as part of common storage.

The underlying concept of simulation can sometimes assume either of two almost inverse forms. A model may have a stochastic flow in a deterministic medium; or, like a percolation process, it may have deterministic flow in a random medium. A simulation may be built from the outside in or element-by-element in

hierarchical combination, as in the formation of social, economic and political organizations.

Comments

The theme of the paper indicates that the HOIST information retrieval simulation is in tune with Greenberger's concepts and ideas on what simulation is all about

The goal is to achieve the maximum fidelity with the minimum complexity, a factor that makes simulation of the information process a most difficult art.

Simulation of 473L System

Originator	Date	Customer	Source of Information
Information Systems Operation of GE	Dec. 1964	USAF AFSC Bedford, Mass.	AD 458001 Simulation and Analysis of 473L System. Vol. I. Simulation of 473L System Over a Range of Operating Conditions

Summary of Method

The overall 473L Command and Control System was modeled and simulated in order to predict performance over a range of operating conditions. Disc access time was the variable of primary interest. Performance was measured in terms of response time, computer and disc utilization, and total storage requirements. An analytic solution was derived which substantiated the simulation results.

The 473L System provides for the gathering, transmitting, processing and display of information to allow the USAF to effectively plan, organize, evaluate and support strategic and tactical moves. It is tied to other systems using AUTODIN, teletype and voice communication facilities. The major hardware groups are a powerful data processor which has priority interrupt features, with the ability to process many functions simultaneously; a content addressable disc memory; a buffer processor to handle the direct link to AUTODIN, the peripheral devices, Unirecord facilities, and the integrated consoles which provide two-way communication between the system and its users and among the users. The system incorporates a structured data base and a large set of programs. Queries are recognized by the system, analyzed for priority ranking and either operated on or stored for further operation.

A model of the 473L System was constructed based on the System Modeling Technique developed by the Information Systems Operation of GE. The model used SIMSCRIPT on an IBM 7090 computer. The following measures of performance were estimated:

1. Average response time to a query dependent on priority (does not include display generation time).

2. Average turnaround time (includes display generation time).

3. Percent utilization of the central processor and the disc.

4. Maximum and average number of words in core and in the real-time storage.

Comments

The conclusions or findings of the simulation were largely positive, that is, in favor of the system as designed and used. Service provided low priority requests was found equivalent to service provided high priority requests. Computer and disc utilization was found to be unbalanced, with the disc being busy as high as 40.45 per cent of the time and the computer only 12.7 per cent of the time. This seems to have been the only negative finding. Recommendations were made to balance utilization and to reduce the complexity of the interrupt scheme mostly to reduce the costs of computer programming and to reduce the number of instructions in core or disc.

SCERT (Systems and Computers Evaluation and Review Technique)

Originator	Date	Customer	Source of Information
COMRESS, Inc.	Spring, 1964	-	AFIPS Conference Proceedings, Vol. 25, 1964 Spring Joint Computer Conference

Summary of Method

Systems and Computers Evaluation and Review Technique (SCERT) is a simulation program which has been designed to accept definitions of a data processing problem and to build a mathematical model of each program run in the defined problem. SCERT maintains a library of hardware and software performance factors for a wide range of digital computers. Using the algorithms which have been incorporated into the program, it can extract the appropriate hardware and software factors for all the components in any one configuration. With this information, it will build a mathematical model representing the hardware-software performance capabilities of each selected computer configuration. During the simulation phase then, SCERT simulates the response of each of the "program models" against the "performance model" of each of the selected configurations.

The results of this simulation are represented in the form of several different management reports which furnish the user with projections of cost, time, memory and manpower requirements which would be necessary to put his data processing system "on the air" for any of the computers evaluated.

Comment

SCERT is an extremely large and complex evaluation program consisting of 31,000 instructions, approximately 5000 algorithms and approximately 100,000 hardware and software factors. It requires two-to-four hours of running time for a problem consisting of 100 computer runs for the evaluation of six different computer configurations. The complete production of the SCERT analysis takes about four weeks. While it is a good technique once the characteristics are well-established, there are no provisions for ranges of values. It provides answers to many problem areas of computer evaluation, but it is data processing oriented rather than information retrieval oriented. SCERT is primarily a management

aid rather than a system engineering aid. It operates as a service group contract, with each evaluation costing approximately the same as one month's computer rental. Because of its size and complexity, it does not permit in-house capability for the user. More hardware than software oriented, SCERT is written only for a large computer (RCA 301),and accepts only fixed or known characters of the hardware, hardware complex or system being evaluated.

Comparison of SIMCOM and SIMSCRIPT Simulation Techniques

Originator	Date	Customer	Source of Information
Information Systems Operation of GE	Dec. 1964	USAF AFSC Bedford, Mass.	AD 458 002 Simulation and Analysis of 473L System, Vol. II. SIMCOM and SIMSCRIPT Simulation Techniques: A Comparison

Summary of Method

A comparison of the efficiency of the SIMCOM and SIMSCRIPT simulation techniques was made based on their application to a common simulation model. The results demonstrated that SIMCOM is more efficient in the initial set-up stage and that SIMSCRIPT is more efficient in the production stage. Curves of cost versus the number of production runs were plotted showing the number of production runs at which the two techniques require the same expenditure of funds.

The advantages of SIMCOM are:

-- least cost for a small number of computer runs (10-15).
-- time to obtain results minimum since the translation of the model is quickest.
-- least cost and difficulty in altering the simulation model.

The advantages of SIMSCRIPT are:

-- least cost for a large number of production runs, greater than 10-15 runs.
-- provisions for the greatest flexibility in output summarization and in the size and complexity of the model.
-- availability on more computers.
-- more widely known and used; easier to obtain information about its characteristics.

Comments

This is a very short report of nine pages, but it contains some interesting charts on costs to perform simulation, relative costs as a function of the number of computer runs (1000 samples to each run) and some indications of the cost of training for simulation. Such figures as these are valuable to the assessment and use of an information retrieval model, and the techniques employed are worth consideration by the model developer and by the system planner or system engineer for whom the model is built.

Project MAC Time-Sharing

Originator	Date	Customer	Author	Source of Information
MIT	(No date given)	--	J. H. Saltzer	AD 612 702, CTSS Technical Notes

SUMMARY

This is a technical description of the 7094 Compatible Time Sharing System in use at Project MAC and the M.I.T. Computation Center. It is written primarily for the programmer, but it contains valuable information for the user (nonprogrammer) of the system. The special features of the 7094, such as the data channel (especially the 7750 Communications Channel), are described and the design principles for input and output through the control of the supervisor are explained.

On the matter of relations among user, user program and supervisor, there are some interesting new terms which pertain to time-sharing systems. Each state of the "logged-in" user has an associated code number in the MAC system, as follows:

0 - "Dead." Corresponds to a user without a program, that is, his program may not be loaded or it may have completed execution and returned him to the "dead" state.

1 - "Dormant." User is in the dormant state when he has a program which is potentially executable but which he does not want to run at the moment. User will be in this state if he has just loaded a program but has not yet started it, or if he has just finished the program and returned to the dormant state (core image of program remains available).

2 - "Working." The program is scheduled for execution. All of the users, although only one is served in a given instant, who are trying to execute are considered to be in the working state.

3 - "Waiting command." When the user is in the dead or dormant state, whatever he types is considered as a command to the supervisor, and he is placed in state 3 for active consideration by the scheduling algorithm. (Note: once loaded, such a command program is indistinguishable from one of the user's programs.)

4 - "Input Wait." When the working user program attempts to read a line of input which is not finished from the console typewriter, the user is assigned to state 4, that is, he is temporarily ignored until the needed line has arrived.

5 - "Output Wait." When the user's program attempts to produce messages at a rate higher than the console can type, intermediate buffers absorb a few such messages and the user is placed in state 5 until the messages clear sufficiently for the program to proceed.

COMMENTS

The heart of the time-sharing system is the scheduling algorithm, which can account for the priorities and perform time integrations on queries in queue according to a particular scheduling policy. Flexibility comes then from the type of scheduling algorithm chosen and from variations permitted within that algorithm once selected.

Data Processing System Simulator (DPSS)

Originator	Date	Customer	Source of Information
Systems Development Corporation	Fall 1964	Project 465L	AFIPS, Fall 1964

Summary of Method

The Data Processing System Simulator (DPSS) is a general purpose computer program that can be used for the evaluation of a proposed new design or a modification to an existing data processing system prior to making equipment selections or performing any significant computer program design.

DPSS uses a higher order simulation language similar to those which have been developed previously for general simulation. However, unlike other techniques, a single combination of these higher order language macro-instructions is used in a single logical arrangement to permit the representation of a wide variety of possible data processing system configurations and processing rules with no additional programming or design.

DPSS has time-sharing features, interrupt capability and batching capability. The language used is JOVIAL, a high order SDC programming language, and there are approximately 1500 instructions. The simulation was initially designed to run on the AN/FSQ-32-V computer, but the latest and expanded version is written for the IBM 7094.

Comments

The DPSS is a time-based event-oriented simulation which requires for input the definition of the program system configuration and the assignment of values for the system parameters. It is limited primarily to on-line multiprocessing simulation of the central processor. One must specify the length of the period to be simulated, the number of times the test is to be repeated under the same operating conditions, the number of each type of message that will arrive, the batch criteria (time, size and interrupt), the system tasks to be done, message-task relationships, message-display relationships, task sequence, traffic rate and other time data. The meaning and types of messages are not explained clearly. Time data produced consists of message arrival times, cycle

times, number of messages, begin- and end-processing time of each message, and other time data. The application is primarily to real-time systems, such as the 465L SACCS. In use, the system was most sensitive to the interrupt feature, which could cause short cycle times to create the impression of highly efficient operation. It was not particularly sensitive to the relative frequency of each message type for a given input rate when all or most types of messages were present. It was noted that there was a rather high proportion of time to load versus time to process, which seems to bear out some I/O limitations of the system operation.

The capabilities of DPSS are expressed in general terms. It is used for system feasibility studies, the simulation of computer-based data processing systems, establishment of equipment configurations, development of design requirement, the setting of initial operational parameters and the determination of system performance requirements. Further developments of DPSS are said to be in the multi- and parallel-processing areas with considerable emphasis upon prediction techniques.

VECTOR

Originator	Date	Customer	Source of Information
Auerbach Corporation	Jan 1, 1964	ESD	AD 435557

Summary of Method

The VECTOR process is a procedure for producing quantitative, objective estimate of the performance of digital computer systems. The essence of the VEC' . process is the production of descriptions, in standardized formats, of the characteristics of each problem to be solved and each computer system to be considered. Furthermore, the method provides a series of synthetic measures of applications, which when adapted to the variations in each computer system design, enables the subjective considerations of the system evaluations to be removed. Then, by means of straightforward calculations, the estimated processing time for a specific application on a specific computer can readily be produced through a standard procedure. Thus, valid comparisons of applications for varied computer systems can be made without fear of one system being favored over the other. At present, it is a manual method.

Comments

During the study, it became obvious to Auerbach that one average figure was not accurate enough, and they finally went to a performance factor scheme. There was also a need to simplify the method of specifying parameters in order to make the process easier to use. A very detailed method of specifying parameters is given.

Reverse (Page 128) Blank

Performance Simulator (PS)

Originator	Date	Customer	Authors	Source of Information
Arthur Andersen and Company	March, 1962	Office of Science Information Service, NSF	Dr. D.B. Hertz, B.E. Wynne, Jr., J.J. Corrigan, Dr. K.H. Schaffir, Mr. L.A. Moody, and Dr. S.B. Littauer of Columbia University	AD 273 115, Research Study of Criteria and Procedures for Evaluating Information Retrieval Systems, Final Rep.

NOTE: This summary applies only to Chapter VI, The Performance Simulation Model, of the report. The report covers a very broad range of problems associated with information retrieval, and only the simulation section is treated here. Performance simulator runs are contained in Appendix A of the report.

SUMMARY

The objective of the simulation was "to faithfully reproduce the input, search and output characteristics of existing or proposed information retrieval systems." While the desired results were to provide a useful tool for evaluating IR systems and to provide a framework by which designers and users can better understand and use any IR System, a compromising position was taken by the authors by saying that at best the effort was exploratory, with the primary purpose being to achieve the feasibility of simulating such a system. It is a "black box" simulation, not a "literal" simulation.

There are five phases to this simulation:

1. Generation of requirements. The statistical descriptions of blocks of these requirements correspond with measure on groups of user needs for the IR system simulated.
2. Specification of search strategy. The generated requirements are translated into requests which serve as object selection criteria in the simulator. Rules governing the strategy relate to real system rules.

3. Creation of file content. Index descriptors and stored objects are statistically equivalent to a real file.

4. Identification of responsive objects. File content is matched with individual request specifications with varying degrees of matching permissible.

5. Evaluation of search results. This is the degree to which descriptor-object intersections given by the requirements are fulfilled by the retrieved objects.

The uses of the performance simulator (PS) fall into four areas of application. The first is very general in that it is an empirical development providing leads for theoretical analysis. Two uses are evaluative, one being cost-time-volume (CTV) studies for actual installation or operation and the other being CTV design studies on prototype or hypothetical IR systems. The fourth application which is visionary or anticipatory is user and/or operator training in lieu of expensive real-time learning.

There are three classes of simulation input. (1) inquiry statistics, which consist of descriptors per inquiry, individual descriptor usage, and distribution of logical inquiry frameworks, (2) search strategy and limitations, (3) file content statistics, which consist of objects per descriptor, number of descriptors per object and size and density of file.

All semantic problems associated with coding, indexing or structure were completely ignored, and so are the errors associated with these areas. It is presumed that the model can be extended to include these.

Inputs are as follows:

Number 1 -- distribution of relative frequency of use of each descriptor.

Number 2 -- distribution of relative frequency of use of specific Boolean statements in expressing user requirements.

Number 3 -- Frequency distribution of objects desired per user query.

These three inputs and random numbers generated in the computer are used to generate user inquiry requirements, which are supposed to be statistically

similar to inquiries in a real system. They are internally generated, and hence, they exist in machinable form in the common language and allow the hypothesizing of quantified changes in the type of user inquiry. A simulation consists of a block of 100, or some other number, of these requirements. Requirements are regarded as a body of knowledge desired by the user of the system being simulated. The descriptors of the matrix represent units of information desired. The output is simply a list of the objects which respond to the request.

COMMENTS

For the user whose interest is statistics on descriptors and usage, search strategy and file content, the performance simulator has something to offer. For the user concerned with the problems of evaluation which relate to structures, indexes, input and output, it is inadequate.

Reverse (Page 132) Blank

Analysis of Serial and Parallel File Structures (473L System)

Originator	Date	Customer	Source of Information
Information Systems Operation of General Electric	Dec. 1964	USAF AFSC Bedford, Mass.	AD 458003 Simulation and Analysis of 473L System, Vol. III. Analysis of Serial and Parallel File Structures

Summary of Method

This study was undertaken to determine if the introduction of data redundancy can reduce search time in storage and retrieval applications that use magnetic tape. Serial and parallel file structures and directories on magnetic tape and disc memories were examined. It was concluded that search time can be improved through utilization of data redundancy and the random-access features of disc memory.

The study was performed to provide design guidance for COLINGO, a single user command and control system developed by the MITRE Corporation using the IBM 1401 computer. The COLINGO system has all data files stored in serial form on magnetic tapes. Each record in a serial file is a logical unit; all items are the properties of the header. A dictionary describes the uniform record format by listing each property name, its relative position in the record, and the number and type of characters representing its value. A query names a particular file and lists any number of that file's property names as qualifiers, logically linked to values. The information requested is retrieved by bringing all candidate serial records into core from off-line storage, identifying relevant items in each record, and testing for the appropriate values. Whenever a serial record is found which satisfies the query, it is written onto an output tape. This tape is not only used for output of the response to the query, but it also comprises a subfile to which subsequent queries may be addressed.

Comments

The equations on file search time are valuable for use in subroutines in simulations which require such expressions for search time. For example, an equation is given for the average number of parallel records searched in response to a query. Time to read one record of length L from tape is given.

Important searching time is saved with redundant parallel file on tape, even for partial parallel file. Time-saving formulas are given, such as the time saved when the partial parallel file is put on disc and the serial file retained on tape, in which case less disc storage space is needed.

Although it is impossible to use a random-access disc for the storage of the partial parallel file, average search time can still be improved by using two tapes.

Reactive Typewriter

Originator	Date	Customer	Author	Source of Information
Zator Company	Oct. 1962	AF Office of Scientific Research, Washington 25, D.C. (AFOSR-2711)	C.N. Mooers	AD 400 349

Summary of Method

Mr. Mooers directs some strong critique at current and fashionable topics in information retrieval. He argues for the future use of remote computers by means of typewriters in the library with wire connection to the computer. The cost should be in the order of $500 to $1000 for the small library users, but actually runs nearer to $10,000

He mentions programming languages for the typewriter and describes the features of the TRAC language, which is Mooers' own text reckoning and compiling language.

To Mooers the information crisis is nothing new because information has been growing rapidly for the past 150 years. Strangely enough, he says one of the really fatal things to happen to many retrieval systems is to have the system finally put into successful operation because the plans and discussions on the system design come to an end. Interest drops and the system dies a slow death.

Comments

The best retrieval machines, according to Mooers, are not affordable, not even by DDC (ASTIA). Examples are the "Minicard" and the "Walnut." Even if these were affordable, they cannot keep track of document inventories, check on "need to know" and do many other essential tasks. The painful problems are the internal library operations -- card typing, filing, replacement on deterioration -- and these "housekeeping" tasks are not touched by the selector machines. Mooers makes a plea to hardware manufacturers to consider these areas, and hence his reactive typewriter plea.

The absence of a measure of retrieval efficiency is a severe malady in retrieval work, according to Mooers. Machines are built by computer engineers

who have less perspective of the problems of readability and use than the librarian or user, with an example being the use of only upper case in computer readouts. He makes the point that the computer engineers are not doing enough about easing the drudgery of bibliographic operations in the library. He predicts that card catalogs will eventually disappear.

There is reason to believe that, according to Mr. Mooers, the problems of evaluating information retrieval systems are related closely to the support (e.g., clerical tasks) required for such systems and to the difficulties (e.g., remoteness) of conveniently using the product of the system. These are factors which must be considered ultimately even though they are somewhat external in any IR system. Mooers wants better integration of support and of use with the retrieval process itself such as would be accomplished with the reactive typewriter which is, of course, now a reality.

Adaptive Techniques in Textual Data Retrieval

Originator	Date	Customer	Author	Source of Information
Astropower Laboratory, Missile and Space Systems Div., Douglas Aircraft Co., Newport Beach, California	Aug. 1964	Information Processing Branch, Rome Air Dev. Center, Research and Technology Division, AFSC, Griffiss AFB, New York	J.A. Daly, V.W. Goldsworthy, Dr. R D. Joseph, E.E. Nelson, Dr. D M. Ramsey, M. Uemura, S.S. Viglione	AD 605 260 Adaptive Techniques as Applied to Textual Data Retrieval

Summary of Model

This is a document covering a broad range of problems in textual data retrieval. It begins with a description of the Astropower Document Screening System, which consists of (a) a word recognizer, (b) a static screening device and (c) an adaptive monitor. The word recognizer includes character recognition, word detection and dictionary look-up. The static screening device uses the keyword output of the word recognizer (including the "values" assigned to them to classify the document), and the adaptive monitor uses the keyword output of the word recognizer and the pertinence decisions of the screening device to generate "values" for the keywords.

Work was performed to find a method for automatically assigning weights to keywords and phrases, and to find the adaptive technique for keeping the static model responsive to changes in the meaning of words and alert to the introduction of new concepts. Lack of correlation for particular words and potential new meanings would be referred to the human operator for re-evaluation.

The different adaptive techniques in data retrieval are described. Five of them are:

1. Forced Learning
2. Error Correction
3. Logarithmic Weights
4. Numerical Design
5. Association Logic.

The adaptive techniques were evaluated and none of them was successful in achieving prediction comparable to the minimum correlation between expert judges. However, there is a small but definite correlation between some of the adaptively designed mechanisms and the humanly designed system, which means (according to Astropower) that obsolescence in an adaptively designed structure might predict obsolescence in the corresponding portions of a screening device. The problems of mechanizing such an obsolescence monitor are discussed.

The methods of implementing adaptiveness fall into three broad areas:

1. analog memory devices
2. special-purpose digital hardware
3. simulation on general purpose digital computer.

Computer simulation is regarded by Astropower as a very powerful approach to adaptive studies. The adaptive element is assigned a memory location; the logic capability determines how the adaption should proceed and the arithmetical functions in the computer implement the adaption. All of the adaptive techniques mentioned previously were handled through digital simulation. Pattern recognition using time-varying threshold logic was employed and formulas and diagrams are given for character recognition, threshold levels, loss functions, input weights and output weights. The program was written in Fortran II. It contains a main program to input the data, call subroutines to perform separate calculations and outputs the program results.

Description of the Simulation Program

The input data to the program specifies the activity pattern of the retina cells for each alphabetic character. The computations of pattern weights (losses) starts the selection of logic units, and a random number generator routine is used to determine the connections from the logic units to the retina. A subroutine (called EARLE) is used to compute the input weights and the activity of the logic units. This subroutine uses another subroutine (OSCAR) and two library routines to calculate the eigenvectors and eigenvalues needed in computing the input weights. Subroutine EARLE also calls subroutine SEX to determine the threshold levels. The main program calls subroutine DAVE

to compute the logic unit output weights and the loss functions. Finally, subroutine DIANE redetermines the pattern weights.

Twenty logic units are generated in this manner before the best one is selected for inclusion in the machine. After a unit is selected, two iterations on subroutines DAVE and DIANE are performed to re-evaluate the machine and respecify the pattern weights and logic unit output weights. Subroutine SAM is then called to tabulate machine errors for print-out.

Flow charts of the computer operation and of the main program and each subroutine are given.

Results of the Simulation Program

The purpose of the time-varying threshold logic structure (TVTL) was to obtain error-free operation on the problem of character recognition. A good degree of success was achieved, e.g., with a 17-logic unit machine as a recognition device, three errors were made in 240 tries for an error rate of 1.25%. The results of this design procedure were compared with other methods in order to evaluate the efficiency of TVTL. A previous design (IDID) was tried using linear logic units and iterative design (numerical design, and iterative minimum loss selection) on both the input and output levels. Another design (DAID) involves linear logic units with discriminant analysis used to specify fixed threshold units and iterative design to approximate the decision hyperplane. (DAID is said to be better than the forced learning procedure in perceptrons for similar problems.)

The IDID machine reached its best performance (minimum loss) with about 32 units. The DAID machine equalled the performance of the IDID machine with 57 units. The 15-unit TVTL machine had about 1/7 the loss of the 32-unit IDID and the 57-unit DAID devices.

Comments

The document because of its detailed descriptions which are supported by many tables and illustrations requires considerable time to read and even more time to evaluate. It is one of the most detailed documents available on the problems of retrieval, particularly the problems of character recognition. It contains considerable hardware data on analog memory elements and on character

readers. Very good appendixes are included: log weight optimizations, the numerical design process, bibliography on document corpora used, a common word list, key word judgments, human judges and their qualifications, and various mathematical equations and analyses. Results of each adaptive technique are fully treated especially with charts and tables and fairly presented in that Astropower gives the negative as well as positive results.

There are no direct applications to the G model, but simulation as a tool in handling adaptive techniques for information retrieval is highly extolled. While the simulation is largely hardware oriented, it is a most valuable technique to obtaining results which indicate the performance of the retrieval process in handling characters. Astropower's simulation which is largely an error-rate process, is document-oriented, and the results must be interpreted accordingly.

Simulation and System Theory

Originator	Date	Customer	Author	Source of Information
SDC	April 1964	Professional paper	Michael R. Lackner	AD 610 697, Digital Simulation and System Theory (to become a chapter in Methodological Developments in Simulation of Organizational Behavior, edited by Harold Guetzkow, 1965

Summary of Method

Human organization systems as object systems of ligital simulation require means of identifying their components, characterizing each one, articulating their interrelationships, stating the whole and drawing conclusions. Nothing short of general system theoretical develop. ~nt will do. Existing simulation schemes do not explicate the theory of the models they are used to build although they do organize a set of solutions to the practical problems of implementing a very complicated computer program. A scheme for modeling processes and producing a well-organized simulation model is suggested, and a calculus of change is introduced as an approach to the general problem of developing system theory.

Comments

The material is written as a textbook and is to be used in such a book. It is, therefore, very instructive and quite comprehensive in scope. The author describes simulation models as belonging to two types: (1) the activity- or process-oriented simulation model, and (2) the event-oriented model. In the first, events are associated with the initiation or completion of processes and in the second, events are identified and characterized as individual algorithms. Figures are given to illustrate in general the difference between the process-oriented model and the event-oriented model. The validation of such models is no longer considered primarily the judgment of the modeler, but rather the "functional validity" of the relationships as used.

Two important terms appear in the process of modeling. An "open model" and a "closed model" are used to distinguish the elements of the model that are modifiable by any process at any time from those modified exclusively by one process.

Some discussion of simulation languages is given. Popular list-processing languages are IPL V by A. Newell (1961); LISP, J. McCarthy (1960), and COMIT, M.I.T. (1962). Languages which deal directly with the problems of parallel activity are Dynamo, Forrester (1958); SIMSCRIPT, Markowitz, Hausner and Karr (1962); General Purpose System Simulator (GPSS), Gordon (1961); General Simulation Program, Laski and Buxton (1962); and SIMPAC, Lackner (1962). The G.P.S.S. requires a model in terms of "transactions" which seize "stores" or "Facilities" and form "queues." SIMSCRIPT requires a breakdown in terms of "entities," "attributes," "sets," and "events." SIMPAC expects a model in terms of "logical situations," "activities" and "transient information."

Short Treatise on Model Design

Originator	Date	Customer	Author	Source of Information
RAND Corp.	April 1965	-	Ira S. Lowry	AD 614 413, A Short Course in Model Design

SUMMARY

The growth of computer models is attributed to the increased sophistication of professional planners and to the inadequacy of present techniques. Little is owed to the proven adequacy of such models. The essay is designed to provide orientation to the model builder's thinking, to interpret the jargon of the trade and to suggest some standards for the evaluation of the model as a product.

Models are divided into three classes: (1) descriptive, (2) predictive, and (3) planning. The accomplishments of the model builder are measured by (1) ratio of input data required by the model to output data generated by the model, (2) the accuracy and cost of the latter as compared to direct observation of the variables in question, and (3) the applicability of his model to other times and places than that for which it was originally constructed.

Planning models are the least developed of the three types. The planning model incorporates conditional prediction, but it goes further because the results are evaluated in terms of the planner's goals. The steps are:

1. specification of alternative programs or actions
2. prediction of the consequences of choosing each alternative
3. scoring the consequences according to a metric of goal-achievement
4. choosing the alternative which yields the highest score.

The strategy of model design is discussed. The macro-analytic approach to model building is critiqued, with the two major objections being the lack of explicit casual structure and the failure to lend themselves easily to financial accounting schemes. The micro-analytic approach is similarly criticized. The first objection is that a model based on rational choice can be implemented only when the chooser's relative value can be specified in considerable detail, and the second problem is the implementation of a model which is comprehensive

enough to embrace the entire range of transactions affecting the world being modeled (e.g., comprehensive market model).

The role of time is not an unexpected point in the essay. The treatment of time varies from degrees of temporal continuity (that is, from comparative statics at one extreme to various types of recursive progression, to analytic dynamics,on the other end). In other words, the author's treatment of time relates to such sophisticated terminology as exogenous and endogenous variables, self-equilibration of relationships and to the other terms which are the jargon of a professional model-builder.

COMMENTS

Since the essay refers consistently to an urban system, which includes the problems of delineating transportation networks, merchandising methods and consumption patterns,it has little direct relationship to an information retrieval model. The strategy of model design, however, is related. The plans for making the model operate, for example, are similar in that concrete steps are necessary from the time input data is fed to the computer until final results are read out. The author gives four methods to achieve this: (1) the analytic solution, (2) the iterative method, (3) machine simulation,and (4) man-machine simulation. The first method is for models which exhibit very tight logical structure, the second is for those models which lack complete logical closure (or possibly burdened with inconvenient mathematical relationships), the third is for loosely articulated systems analyses of lesser mathematical and logical rigor, and the fourth allows for periodic interruption so that an intermediate state of the system can be examined or influenced by a human participant.

The language of simulation continues to be refined as additional simulations are performed. The author refers to "fitting the model," that is, the calibration of the model to a truer simulation. There are no canons for fitting variables but there are some scattered principles to be observed, such as lessening the sensitivity of the model to bad data. Mr. Lowry makes the fitting of the model analogous to the manufacture and assembly of a new piece of electrical machinery.

The final discussion of the model concerns testing which depends on its predetermined function with the descriptive variety being the easiest to test. For a predictive model, it is necessary to run a prediction and verify the details

of the outcome. Testing a planning model involves two phases: (a) a check on its ability to trace through the consequences of a given planning decision (or decisions), described as a form of conditional prediction, and (b) a check on the ability of the model to select an optimal result from a spectrum of alternative outcomes. An alternative to these methods is "sensitivity testing," but the author says this indicates more about the "strength" of the model's design than it does about the descriptive, predictive or evaluative accuracy. Sensitivity testing involves varying the value of a single parameter or input variable in successive runs with the difference in outcome to be measured in association with a given parametric change.

The author's final words are not too encouraging for the model-builder. By and large, the model is a tool of unknown efficacy for the client, and even the best tests the client can insist upon are partial and indecisive; and those who sponsor model-building rarely have the time or training to participate, even in testing. Hence, their interest in the model is secondary. The process of model-building, however, is educational, valuable and worthwhile for the future if not for the present so long as it is not treated as a magic box for yielding answers.

Reverse (Page 146) Blank

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA · R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1 ORIGINATING ACTIVITY (Corporate author)

HRB-Singer, Inc.
State College, Pennsylvania

2a REPORT SECURITY CLASSIFICATION

Unclassified

2b GROUP

3 REPORT TITLE

AN INFORMATION RETRIEVAL SYSTEM MODEL

4 DESCRIPTIVE NOTES (Type of report and inclusive dates)

5 AUTHOR(S) (Last name, first name, initial)

Blunt, Charles R.

6 REPORT DATE	7a TOTAL NO OF PAGES	7b NO OF REFS
October 1965	150	23

8a CONTRACT OR GRANT NO

Nonr 3818 (00)

b PROJECT NO

9a ORIGINATOR'S REPORT NUMBER(S)

352.14-R-1

9b OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10 AVAILABILITY LIMITATION NOTICES

None

11 SUPPLEMENTARY NOTES

12 SPONSORING MILITARY ACTIVITY

Information Systems Branch
Office of Naval Research

13 ABSTRACT

This report presents some work pertinent to the quantitative evaluation of information retrieval (IR) systems and extends the development of an Information Retrieval System Simulation Model. The work is sponsored by The Information Systems Branch of the Office of Naval Research and is part of a program whose objective is to formulate general purpose simulation models of the various functional components found within the intelligence systems.

The present IR system simulation examines system response time, equipment/personnel utilization and idle time, and delay time in queue. The general IR model is, essentially, an ordered grouping of basic retrieval functions. The nature of the functions and the configuration of the system can be specified to the simulation program by the investigating engineer. The simulation program can ultimately be used by Naval planners of information retrieval systems to evaluate alternative IR configurations, to identify and illustrate the need for an IR system, and to assess the effectiveness of such a configuration.

Response time of an IR system is cited as one necessary criterion of system performance and is closely related to the operating costs, another quantitative measurement of an IR system.

DD FORM 1 JAN 64 1473

UNCLASSIFIED

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Information Retrieval, Simulation						
Data Storage Systems						
Models						
Evaluation						
Effectiveness						

INSTRUCTIONS

1. **ORIGINATING ACTIVITY**: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2a. **REPORT SECURITY CLASSIFICATION**: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. **GROUP**: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. **REPORT TITLE**: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. **DESCRIPTIVE NOTES**: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. **AUTHOR(S)**: Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. **REPORT DATE**: Enter the date of the report as day, month, year, or month, year. If more than one date appears on the report, use date of publication.

7a. **TOTAL NUMBER OF PAGES**: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. **NUMBER OF REFERENCES**: Enter the total number of references cited in the report.

8a. **CONTRACT OR GRANT NUMBER**: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. **PROJECT NUMBER**: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. **ORIGINATOR'S REPORT NUMBER(S)**: Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9b. **OTHER REPORT NUMBER(S)**: If the report has been assigned any other report numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. **AVAILABILITY LIMITATION NOTICES**: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through _____."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through _____."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through _____."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. **SUPPLEMENTARY NOTES**: Use for additional explanatory notes.

12. **SPONSORING MILITARY ACTIVITY**: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13 **ABSTRACT**: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS)*, *(S)*, *(C)*, or *(U)*.

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14 **KEY WORDS**: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, rules, and weights is optional.